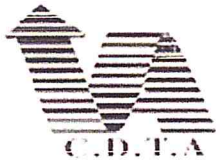
Université Saad Dahlab Blida1



Faculté des Sciences

Département d'Informatique

Mémoire présenté par :

SARDI Zakaria ZAOUI Meftah

En vue d'obtenir le diplôme de MASTER

Domaine : Mathématique et Informatique

Filière : Mathématique et Informatique

Spécialité : Informatique

Option : Génie des systèmes informatiques

Sujet :

Conception et réalisation d'un système d'interaction vocale pour la réalité augmentée

Soutenue le : 29/06/2016, devant le jury composé de :

Mme. AROUSSI Sana	Présidente
Mr. KAMECHE Abdallah Hicham	Examinateur
Mr. ZAIR Mustapha	Promoteur
Mr. BENBELKACEM Samir	Encadreur

Promotion : 2015/2016

Remerciements

Tout d'abord, louange à « Allah » qui nous a guidés sur le droit chemin tout au long du travail et nous a inspirés les bons pas et les justes reflexes. Sans sa miséricorde, ce travail n'aura pas abouti.

Nous tenons à remercier sincèrement nos encadreurs Mr. BENBELKACEM Samir et Mr. BELLARBI Abdelkader ainsi que notre promoteur Mr. ZAIR Mustapha, pour leur aide précieuse et pour le temps qu'ils nous ont consacré. Sans oublier bien sur de remercier Mme. KENOUI Mouna pour ses orientations et ses critiques positives.

Nous n'oublions pas nos parents pour leur patience illimitée, leur encouragement continu, leur aide, en témoignage de notre profond amour et respect pour leurs grands sacrifices.

Enfin, nous adressons nos remerciements aux membres du jury pour avoir accepté d'évaluer ce travail et pour le temps consacré à la lecture de ce mémoire.

Merci à vous.

Dédicaces

A mes chers parents

A mes grands parents

A mes tantes

A mes sœurs

A mes oncles Abd El Karim, Djamel, Samir, Fouad,

Yassine et Hamid

A mes cousins Hamza et Taki Eddine

A mes chers amis Sidaali, Hacen, Yousef, Mohamed,

Yassine et Khaled

A tous mes amis de l'USDB

Je vous dédie ce modeste travail.

Zakaria

Dédicaces

Je dédie ce travail

A toute personne qui sa présence dans ma vie m'est très chère, et particulièrement mes chers parents

Meftah

ملخص

سمح تطور التكنولوجيا في ظهور مفاهيم جديدة، لعل من أبرزها مفهوم الحقيقة المعززة، التي سرعان ما صنعت لنفسها مكانا في مجتمع البحث العلمي من خلال المحاور الهامة التي تعالجها.

أصبح التفاعل مع بيئة الحقيقة المعززة واحد من اهتمامات العلماء والصناعيين، يدرس الباحثون حاليا الأساليب التي تسمح للمستخدمين بالتفاعل بطريقة طبيعية باستخدام قدراتهم الحسية مثل الصوت والإيماءات.

الغرض من مشروع تخرجنا هو تصميم وتنفيذ نظام قادر على التعرف على الأوامر الصوتية للتحكم في الأجسام ثلاثية الأبعاد في بيئة الحقيقة المعززة.

فيما يتعلق بالتطبيق، الهدف هو السماح للمستخدم بتجميع مكونات السيارة، لتحقيق هذا الهدف، أولا اعتمدنا نهجا لتعزيز مشهد حقيقي بجميع المكونات التي تشكل سيارة، ثم استعملنا خادما يستند في عمله على واجهة برمجة التطبيقات للتعرف على الكلام الخاصة بويندوز، للتحكم في هذه المكونات.

الكلمات الرئيسية : الحقيقة المعززة، ARToolKit ،الأوامر الصوتية، التعرف على الكلام، جسم ثلاثي الأبعاد، التفاعل.

Résumé

L'évolution des technologies a permis l'émergence de nouveaux concepts. La réalité augmentée (RA) en fait partie, et elle a très vite retrouvé sa place dans la communauté scientifique en ouvrant un axe de recherche assez important.

L'interaction avec l'environnement de RA est devenue l'une des préoccupations des scientifiques et industriels. Les chercheurs se penchent vers des approches permettant aux utilisateurs d'interagir d'une façon naturelle en utilisant leurs capacités sensorielles telles que la voix et les gestes.

L'objectif de notre projet de fin d'étude consiste à concevoir et à réaliser un système capable de reconnaitre des commandes vocales afin de contrôler des objets 3D dans un environnement de RA.

En terme applicatif, le but est de permettre à l'utilisateur de pouvoir opérer à l'assemblage des composants d'une voiture. Pour atteindre cet objectif, d'abord nous avons adopté une démarche, pour augmenter une scène réelle par l'ensemble des composants constituant une voiture. Ensuite nous avons opté pour l'emploi d'un serveur dédié se basant dans son fonctionnement sur l'API de reconnaissance vocale native de Windows, pour contrôler ces composants.

Mots clés **:** Réalité augmentée, ARToolKit, commande vocale, reconnaissance vocale, objet 3D, interaction.

Abstract

The evolution of technology allowed the emergence of new concepts, augmented reality (AR) take part of this new technology area, it quickly found its place in the scientific community by opening a sizeable research axis.

The interaction with the augmented reality environment has become a concern of scientists and industrialists. Researchers are looking to approaches that allow users to interact in a natural way by using their sensory abilities such as voice and gestures.

The objective of our final project study is to design and realized a system able to recognize voice commands for controlling 3D objects in an augmented reality environment.

In terms application, the purpose is to allow the user to be able to operate the assembly of components of a car. To achieve this goal, first we have adopted an approach to augment a real scene by all the components constituting a car. Then we opted for a dedicated server, which is based in its functioning on the native speech recognition API of Windows to control these components.

Keywords **:** Augmented reality, ARToolKit, voice command, voice recognition, 3D object, interaction, voice command.

Table des matières

Chapitre III : Conception du système

Chapitre IV : Implémentation de système et tests

Liste des figures

Liste des tableaux

Introduction générale

1. Contexte général

L'évolution des technologies a permis l'émergence des nouveaux paradigmes d'interface homme-machine telles que la réalité augmentée. L'interaction avec cet environnement est devenue l'une des préoccupations des chercheurs.

Plusieurs approches pour l'interaction en réalité augmentée sont proposées. Parmi ces approches, nous trouvons des techniques basées sur la reconnaissance de la parole, les gestes, les émotions et l'activité cérébrale de l'être humain.

Notre projet de fin d'étude consiste à développer une approche d'interaction en réalité augmentée basée sur la commande vocale. Ce travail s'inscrit dans le cadre du projet de recherche intitulé « Interaction 3D collaborative et mobile dans un environnement de réalité virtuelle et augmentée » initié au sein de l'équipe Interaction Homme-Machine, Réalité Virtuelle et Augmentée (IRVA) au niveau du Centre de Développement des Technologies Avancées (CDTA).

2. Problématique

La plupart des systèmes de réalité augmentée actuels utilisent des approches d'interaction traditionnelles basées sur le clavier, l'écran et la souris. Les chercheurs se penchent vers des approches permettant aux utilisateurs d'interagir d'une façon naturelle en utilisant leurs capacités sensorielles telles que les gestes, la voix, etc. Notre travail s'inscrit dans cette thématique et nous investiguons l'usage de la voix comme moyen d'interaction 3D dans un environnement de réalité augmentée.

3. Objectif

L'objectif de notre travail consiste à concevoir puis réaliser un système d'interaction en réalité augmentée en se basant sur la reconnaissance de la voix. Ce système devra être capable de reconnaitre des commandes vocales afin de contrôler des objets 3D en temps réel. Les utilisateurs peuvent manipuler des objets 3D en utilisant des commandes vocales simples.

En terme applicatif, nous avons ciblé le domaine de l'assemblage assisté par ordinateur. Ainsi, le but général de notre application est de permettre à l'utilisateur l'assemblage de voiture dans un environnement de réalité augmentée.

Le système que nous développons est décomposé en deux parties principales, à savoir l'enrichissement de l'environnement réel (appelé augmentation) et l'interaction avec cet environnement. Du point de vue applicatif, la partie augmentation concerne l'enrichissement de la scène réelle par l'ensemble des composants (porte, roue, ...) constituant une voiture, tandis que la deuxième partie comporte principalement un module de commandes vocales simples pour le control de ces composants.

4. Organisation du mémoire

Afin d'atteindre l'objectif cité auparavant, nous avons organisé ce mémoire en quatre chapitres :

Le premier chapitre présente une étude bibliographique sur la réalité augmenté.

Le second chapitre présente les techniques d'interaction en réalité augmentée, une étude comparative entre ces techniques est ensuite présentée dans ce chapitre.

Le troisième chapitre est dédié à la conception de notre système d'interaction 3D basé sur la reconnaissance de la voix. Pour cela, nous avons suivi une démarche de conception utilisant le processus 2TUP avec le langage UML comme support de modélisation.

Le dernier chapitre aborde la mise en œuvre de notre système. Nous commençons par effectuer un ensemble de choix techniques sur les ressources utilisées ainsi que sur les architectures. Dans ce cadre, une architecture adaptée à l'application a été proposée, des modules ont été par la suite implémentés. Nous avons ensuite ciblé durant les tests un domaine d'application avec lequel un scénario d'interaction vocale a été mis en œuvre.

Enfin, nous clôturons ce mémoire par une conclusion générale et quelques perspectives des travaux futurs.

Chapitre I :

Etude bibliographique sur la réalité augmentée

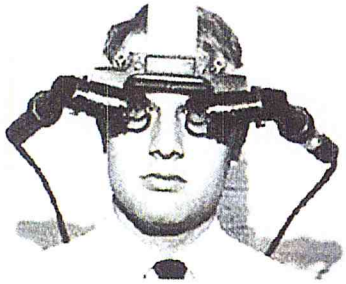
1. Introduction

Nous nous intéresserons dans ce chapitre à introduire et à définir les concepts qui nous permettent de bien comprendre la notion de réalité augmentée. Ainsi, ce chapitre s'attache tout d'abord à définir la réalité augmentée et la réalité virtuelle, puis nous clarifions la relation entre ces deux notions, ensuite nous donnons une description d'un système de réalité augmentée avant de parler des différentes méthodes utilisées pour la localisation en réalité augmentée. Enfin, pour conclure, nous passons en revue quelques domaines d'applications de réalité augmentée.

2. Bref historique

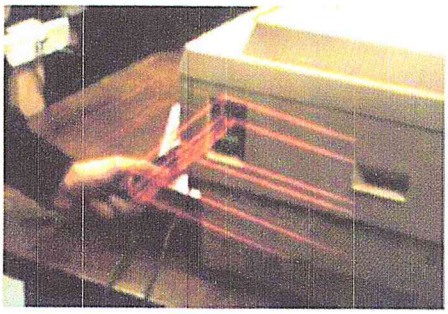
La Réalité Augmentée communément abrégée RA, a vu le jour avec les travaux de Sutherland [SUT, 65], [SUT, 68] (figure 1.a) qui a réalisé le premier système dit de RA, basé sur un casque suivi par un capteur de mouvement. Avec ce dispositif, l'utilisateur pouvait alors visualiser et naviguer autour d'éléments virtuels positionnés dans l'espace réel. Durant les années 80, le concept de RA a été surtout utilisé dans un cadre militaire pour l'affichage d'informations virtuelles sur les pare-brise des cockpits d'avion (Head-Up Display, HUD).

Le véritable essor de ce domaine a débuté durant les années 90 par les travaux de Bajura [BFO, 92], puis State [SLH, 96], pour donner au médecin la possibilité de visualiser directement des données d'imagerie à ultrasons sur le corps du patient, augmentant alors sa perception (interaction et visualisation dans un même espace) (figure 1.b). Feiner [FMS, 93] propose quant à lui un système interactif pour l'apprentissage et la maintenance d'une imprimante basée sur la superposition d'informations virtuelles (de démontage) sur un casque et grâce à une imprimante équipée de capteurs (figure 1.c). Par la suite, la RA s'est développée en introduisant un grand nombre d'applications cibles : médecine (figure 1.d) [GLW, 96], architecture (figure 1.e) [AUG, 15], design, robotique, loisirs, maintenance.



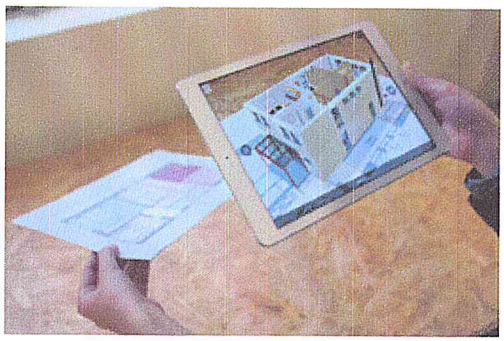
a) Premier casque de RA

b) vue d'un medecin lors d'une échographie



c) vue d'un utilisateur avec un affichage virtuel du casier à papier d'une imprimante

d) Projection de données médicales sur un patient



e) Visualisation augmentée d'un Plan

Figure 1 : *Réalité augmentée de Sutherland à aujourd'hui [SUT, 65], [BFO, 92], [FMS, 93], [GLW, 96], [AUG, 15].*

3. Définition de la réalité augmentée

Trouver une définition précise et consensuelle de la RA n'est pas une tâche facile. En effet, la notion de RA couvre plusieurs définitions :

La RA est définie par Wendy Mackay [MAC, 96], comme une manière de réintégrer l'information électronique dans le monde physique. Il s'agit de permettre aux gens de tirer parti de leurs compétences dans l'interaction avec le monde de tous les jours, tout en profitant de la puissance des réseaux informatiques.

Selon Robinett [ROB, 92], la RA est vue comme un moyen d'augmenter les sens de l'utilisateur, de transformer des événements imperceptibles en phénomènes visibles, audibles ou touchables. C'est vue comme une symbiose entre l'homme et la machine comme jamais auparavant.

Pour Azuma [AZU, 97], la RA autorise l'utilisateur à voir le monde réel avec des objets virtuels superposés au dessus ou composés avec le monde virtuel. La RA complète la réalité, plutôt de la remplacer complètement. Idéalement, il apparaît à l'utilisateur que les objets virtuels et réels coexistent dans le même espace (comme illustré dans la figure 2). Tout système de RA satisfasse les règles fondamentales suivantes :

- Il combine des objets réels et virtuels dans un environnement réel.
- Il fonctionne de manière interactive, en temps réel.
- Il fait coïncider les objets réels avec les objets virtuels.

Fuchs [PFM, 06] quant à lui : la RA regroupe l'ensemble des techniques permettant d'associer un monde réel avec un monde virtuel, spécialement en utilisant l'intégration d'Images Réelles (IR) avec des Entités Virtuelles (EV) : images de synthèse, objets virtuels, textes, symboles.

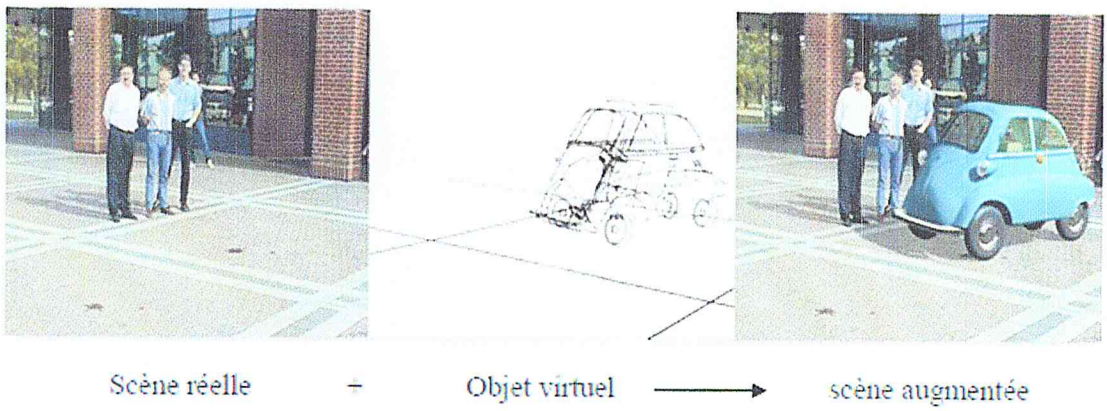


Figure 2 : *Exemple de la réalité augmentée.*

4. Définition de la réalité virtuelle

Pour [PFM, 06], La finalité de la RV est de permettre à une personne ou plusieurs, une activité sensorimotrice et cognitive dans un monde artificiel, crée numériquement, qui peut être imaginaire, symbolique ou une simulation de certains aspects du monde réel.

5. Le continuum de réalité-virtualité

L'évolution des technologies de la RV permet d'immerger complètement l'utilisateur dans un environnement virtuel. Toutefois, cette immersion ne permet pas à l'utilisateur de percevoir et d'interagir avec son environnement réel [MAR, 08]. La RA est donc parfois définie comme prenant le contre-pied de cette approche immersive, et au lieu d'enfermer les personnes dans un monde artificiel, se propose d'utiliser l'ordinateur pour simplement augmenter le monde réel des utilisateurs [WEL, 93].

Afin d'unifier la notion de RA et RV, Paul Milgram [MIK, 94] a défini un continuum de réalité-virtualité présenté dans la Figure 3, dont les deux extrémités de ce continuum correspondent à la réalité et à la virtualité pures, et au milieu on y trouve la réalité mixte (RM), cette dernière représente l'intervalle entre le réel et le virtuel.

La RM contient la RA, mais aussi la virtualité augmentée (VA), qui concerne l'insertion d'objets réels dans un environnement virtuel. La RA et la VA se distinguent par leurs environnements dominants.

Figure 3 : *Continuum de Milgram [MIK, 94].*

6. Système de réalité augmentée

D'après [DOM, 09], un système de RA comprend trois composantes essentielles, comme illustre la Figure 4 :

- une base de connaissances.
- des capteurs.
- des dispositifs de restitutions.

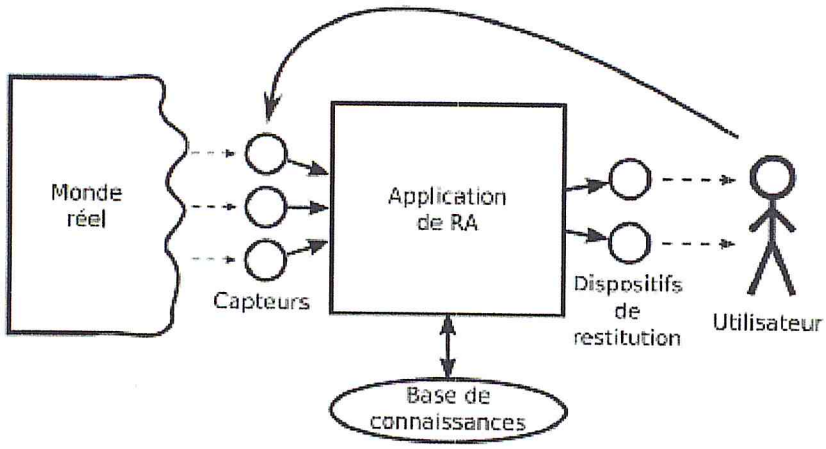


Figure 4 : *Vue globale d'un système de Réalité Augmentée [DOM, 09].*

La base de connaissances fournit une connaissance a priori sur l'environnement dans lequel évolue l'utilisateur.

Les bases de connaissances ne sont pas le moyen unique de fournir une information sur l'environnement. En effet, un ensemble de capteurs peut être utilisé pour fournir une connaissance complémentaire de l'environnement de travail. Il s'agit généralement de capteurs de localisation (ex. Système de positionnement global (GPS), Radio Frequency Identification (RFID), Capteurs inertiels, caméra, kinect) qui permettent de connaître la position et/ou l'orientation de l'utilisateur dans l'environnement.

A partir de ce que fournit la base de connaissance ainsi que les capteurs, l'application RA établit une vue augmentée. Cette dernière est retournée aux dispositifs de restitution, appelés aussi dispositifs de visualisation [ZEN, 10] qui peuvent être des vidéoprojecteurs, des tablettes et des casques de visualisation.

7. Localisation basée vision en réalité augmentée

Les problèmes d'alignements sont un challenge pour les chercheurs, la connaissance de la position et de l'orientation du point de vue de l'utilisateur (de la caméra) est nécessaire dans les applications de RA, car elle permet d'assurer la cohérence de la scène augmentée. En effet, l'estimation des paramètres de localisation permet de modéliser une caméra virtuelle grâce à laquelle un rendu du monde virtuel est réalisé avec les mêmes caractéristiques que la caméra réelle, permettant d'aligner

correctement les mondes réel et virtuel et ainsi de créer la vue augmentée. L'estimation de la position et de l'orientation de la caméra est appelée, estimation de pose [ZEN, 10].

D'après [DEH, 08], l'estimation de pose peut être décomposée en deux étapes :

- Une étape de traitement d'images, visant à obtenir des mesures du mouvement de la scène ou des indices de la position d'un objet.
- Une étape d'estimation proprement dite utilisant les mesures précédentes pour déterminer la position.

L'estimation de pose revient à recouvrir les caractéristiques extrinsèques de la caméra, en se basant sur des modèles de projection, le plus connus est le modèle sténopé (cf. annexe A). Selon le degré de connaissance dont nous disposons de la scène (connaissance totale, partielle ou nulle de l'environnement) nous pouvons regrouper les méthodes d'estimation de pose en deux classes :

Les approches avec ou sans connaissance a priori.

7.1 Approches avec connaissance a priori

Les méthodes d'approche avec connaissance a priori mettant en relation la scène réelle avec l'image courante de celle-ci. Selon la nature des informations extraites, les approches existantes peuvent être classées en deux catégories : la première se base sur des marqueurs artificiels placés sur la scène alors que la seconde exploite des informations naturelles qui existent dans la scène.

7.1.1. Méthodes basées marqueurs :

Dans le but de faciliter l'estimation de pose, plusieurs systèmes de RA utilisent des marqueurs artificiels placés dans la scène réelle. Ces marqueurs sont particulièrement simples à détecter et peuvent contenir un code, ainsi, il est facile de les distinguer. Partant du principe que la position des marqueurs dans le repère monde est connue a priori, l'estimation de pose se décompose en plusieurs phases :

- Détection et identification des marqueurs : cela consiste à extraire de l'image les marqueurs visibles par la caméra et de les identifier,
- Mise en correspondance 2D/3D : à chaque marqueur identifié dans l'image est associée sa position 3D.

- Calcul de la pose de la caméra : estimation de la pose à partir des appariements 2D/3D [ZEN, 10].

Nous pouvons distinguer des marqueurs de deux natures différentes : les marqueurs ponctuels, qui fournissent une correspondance entre un point 3D de la scène et son image 2D et les marqueurs planaires [DEH, 08].

Les marqueurs ponctuels proposés utilisent le plus souvent des formes géométriques : circulaires ou elliptiques. Dans [HNL, 96], Hoff et al ont utilisé des marqueurs CCC (Concentric Contrasting Circle) qui sont formés de cercles noirs sur un fond blanc.

Les marqueurs planaires sont devenus populaires, notamment après l'avènement de la bibliothèque ARToolKit (cf. annexe B). Les marqueurs utilisés dans cette bibliothèque ont des bords noirs sur un fond blanc et un code permettant de les identifier. La figure 5 représente quelques exemples de marqueurs d'ARToolKit.

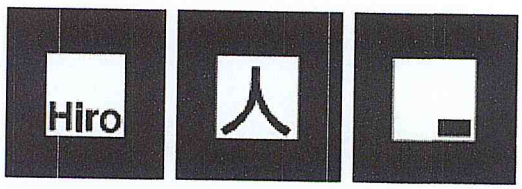


Figure 5 : *Exemple de marqueurs utilisés dans ARToolKit [KAB, 99].*

Les méthodes basées sur les marqueurs sont assez précises. Néanmoins, leurs inconvénient principal est qu'il faut disposer les marqueurs dans la zone de visibilité de la camera, ce qui est inadapté pour les environnements à grande échelle essentiellement en extérieur [DEH, 08].

7.1.2. Méthodes sans marqueurs ou "Markerless"

Les méthodes sans marqueurs représentent une alternative à l'utilisation des marqueurs artificiels en exploitant les caractéristiques naturelles existantes dans la scène réelle telle que des coins, des contours, des segments de droites, etc. Ces approches utilisent des modèles 3D qui constituent une connaissance a priori de l'environnement. Les données 2D extraites de l'image de la scène sont mises en correspondance avec les données 3D [ZEN, 10].

Les méthodes utilisant les contours, sont les plus utilisées [ZEN, 10], la majorité des méthodes dans cette catégorie applique une recherche locale des contours de l'image depuis des points de contrôle répartis sur les contours d'un modèle 3D de l'objet projeté dans l'image.

Les méthodes utilisant les contours sont bien adaptées aux objets comportant des arêtes saillantes : salles, pièces, etc. Cependant, pour les objets naturels trop peu d'informations de contours sont généralement disponibles. Il est alors nécessaire de prendre en compte la texture locale de l'objet, typiquement échantillonnée en des points d'intérêts [DEH, 08]. Nous pourrons se référer à l'article [LVT, 03], où les auteurs proposent une méthode basée sur les points d'intérêts.

Une autre classe de méthodes est celle basée sur les motifs texturés. En effet, elles utilisent des algorithmes qui prennent un motif de référence, et suivent son déplacement entre deux images, dans ce contexte, nous pourrons se référer aux articles : [BLK, 81; BEM, 04].

Chacune des méthodes précédentes a des intérêts et des limites spécifiques, les méthodes basées sur les contours ont l'avantage d'être efficaces et robustes aux changements d'illumination, mais elles sont inadaptées aux scènes très texturées. En revanche, les méthodes basées sur les textures sont robustes aux occultations mais restent sensibles aux changements d'illumination. Ces deux limitations ont fait émerger une nouvelle catégorie des méthodes dites hybrides d'où ses méthodes combinent différentes primitives visuelles, tels que les contours et les textures [ZEN, 10]. Dans ce contexte, nous pouvons citer les travaux dans : [VLF, 04; PRM, 05; PRM, 06].

7.2. Approches sans connaissance a priori

Lorsqu'on travaille sur un environnement a grande échelle, il est parfois quasi impossible d'avoir une vue globale de la scène, les approches dites sans connaissance a priori supposent une connaissance partielle de l'environnement, telle que les techniques appelées SFM (Structure From Motion) [NIS, 03] et SLAM (Simultaneous Localization And Mapping) [MLD, 06] utilisées en robotique mobile. Ces techniques ont l'avantage d'estimer la pose de la caméra et de reconstruire un modèle partiel de l'environnement. De telles approches permettent d'envisager et d'utiliser des modèles 3D partiels de l'environnement et de les enrichir en ligne en reconstruisant les parties non-modélisées [ZEN, 10].

8. Domaines d'application de la réalité augmentée

La RA retrouve ses applications dans plusieurs domaines, on pourra citer notamment les applications potentielles dans les domaines suivants : médical, architecture, industrie/maintenance, robotique, éducation, etc. Nous donnons ici un aperçu des domaines d'application des systèmes de RA.

8.1. Médical

Un des domaines les plus prometteurs est celui de l'assistance au geste chirurgical. Les systèmes de RA peuvent être utilisés lors des opérations pour superposer des données médicales sur le corps du patient. Les données superposées peuvent être directement issues de radiographies ou d'images IRM ou bien être un modèle 3D reconstruit à partir d'une succession de scans. Le chirurgien possède alors une vision étendue des organes sans avoir recours à des techniques plus intrusives. Sato et al [SNT, 98] proposent un système permettant la superposition sur un flux vidéo d'une tumeur cancéreuse dont un modèle 3D est reconstruit à partir d'images ultrasons.

La RA peut être aussi utile pour l'apprentissage et l'entraînement de futurs chirurgiens, les différentes étapes d'une opération pouvant être superposées à la vue de l'apprenti. Dans ce contexte nous pouvons citer le système Medarpa [MED, 03].

8.2. Éducation

La visualisation et simulation 3D en temps réel dans un environnement familier facilite la compréhension des étudiants de concepts difficilement interprétables en 2D. La RA est exploitée comme un outil pédagogique pour susciter l'intérêt des étudiants et stimuler leur créativité à travers une expérience ludique et mémorable [CHC, 11].

8.3. Maintenance et assemblage industriel

Les tâches d'assemblage et de maintenance des systèmes et des équipements complexes représentent l'un des domaines cibles de l'utilisation de RA, les principales perspectives attendues de cette utilisation sont le gain en productivité et une réduction des taux d'erreurs. Ainsi, les instructions de montage et de réparation apparaissent sous forme des modèles 3D superposés sur la pièce à manipuler exprimant les étapes de travail une par une. Les coûts, parfois importants, de maintenance (stockage et mise à jour) des manuels papiers sont également réduits [ABD, 13].

Plusieurs travaux ont montré l'apport des systèmes de RA, pour la productivité dans les chaînes de montages et pour la maintenance et la réparation dans des environnements industriels. Parmi ces travaux nous pouvons citer celui de KARMA [FMS, 93] (Knowledge-based Augmented Reality for Maintenance Assistance) qui est un prototype du laboratoire expérimenté par l'équipe de Freiner au début des années 1990, l'objectif de ce prototype est la maintenance d'une imprimante laser.

8.4. Robotique

En robotique, l'intérêt dépend du domaine d'application, L'idée pourrait être de s'entrainer à manipuler un robot chirurgical grâce a la RA, avant de passer à une opération réelle [LEB, 11].

[IHU, 15] parle d'une chirurgie robotisée guidée par la RA sur une patiente atteinte d'une tumeur au foie. Après une simulation des déformations de la paroi abdominale, le positionnement des instruments chirurgicaux a été défini, puis la RA a été intégré dans la caméra stéréoscopique du robot. L'interaction constante entre le chirurgien et un ingénieur informatique a permis une navigation à vue dans les structures anatomiques, rendant le foie transparent avec visualisation de tous les vaisseaux sanguins normalement invisibles.

8.5. Architecture

Dans le domaine de l'architecture, il est possible de créer des maquettes virtuelles plus écologiques et économiques car il n'y pas d'achat de matériaux. Une maquette de ce genre est beaucoup plus interactive et on peut tout de suite changer un aspect non apprécié, pas besoin de tout refaire. On peut aussi faire une coupe dans la maquette pour visualiser l'intérieur et donc effectuer un meilleur agencement car avec un tel outil, on se rend compte de ce qui ira ou pas en réalité. Sur des plans, on n'a pas la même approche et on ne voit pas directement l'objet réalisé [HAG, 12].

9. Conclusion

La RA présente un domaine où le potentiel d'exploitation est bien réel, notamment dans le domaine médical et éducatif. Au fil de ce premier chapitre, nous avons abordé quelques définitions de la RV et RA, nous avons également présenté la vue globale d'un système de RA. Dans un second temps, nous avons mis l'accent sur quelques défis que rencontrent la RA et plus particulièrement l'estimation de la position et de l'orientation de la caméra, nous avons parlé aussi sur les méthodes utilisées pour l'estimation de pose. Enfin nous avons présenté les domaines d'application de la RA. Dans le chapitre suivant, nous nous intéresserons aux différentes techniques d'interaction existantes dans la RA.

Chapitre II :

Techniques d'interaction en réalité augmentée

1. Introduction

Un des intérêts principaux de la RA consiste à fournir à l'utilisateur des méthodes novatrices lui permettant d'interagir avec le monde augmenté. Dans ce chapitre nous présentons les notions d'interaction et de modalité, nous présentons également les différentes techniques d'interaction utilisées dans la RA et nous concluons par une comparaison entre ses techniques d'interaction.

2. Définition d'une interaction

Selon Larrue [LAR, 11], l'interaction représente un mode de communication entre l'homme et la machine permettant la réception et/ou l'émission de l'information.

Coomans et Timmermans [COT, 97] désignent l'interaction comme la capacité mutuelle entre le système et l'utilisateur d'agir et de répondre aux actions de chacun.

Une interaction, dans le langage courant, est l'action ou l'influence réciproque qui peut s'établir entre deux objets ou plus. Une interaction est toujours suivie d'un ou plusieurs effets [7].

3. Définition d'une modalité

L'utilisateur interagit avec le système informatique pour réaliser des tâches. Les moyens d'action et de perception, appelés « modalités d'interaction », sont les médiateurs matériels et logiciels permettant à un utilisateur d'agir sur le système informatique ou de percevoir son état. Les modalités d'interaction composent le module interface du système informatique. Nous pouvons distinguer les modalités d'interaction en entrée (moyens d'action de l'utilisateur pour interagir avec le système informatique) des modalités d'interaction en sortie (moyens de perception de l'état du système informatique par l'utilisateur) [JBO, 06].

4. Les principales tâches d'interaction dans la RA

Il n'est pas évident de séparer les techniques d'interaction spécifiques aux environnements de RA de celles utilisées en RV puisque un grand nombre de ces techniques sont valables dans les deux types d'environnements [HAY, 11].

En 1999, Bowman [BOW, 99] a proposé une classification moderne des différentes techniques d'interaction selon quatre tâches principales :

- **Navigation** : consiste à explorer et à examiner le monde virtuel.
- **Sélection :** consiste à désigner de façon explicite un objet particulier dans le monde virtuel, afin d'effectuer une action donnée.
- **Manipulation :** consiste à effectuer une certaine action sur un objet donné (déplacement, translation et/ou rotation).
- **Contrôle d'application :** appelée aussi tâche de commande. Elle consiste à exécuter une commande dans le but de changer l'état du système ou le mode d'interaction.

5. Techniques d'interaction les plus utilisées dans la RA

Plusieurs techniques d'interaction en RA ont été proposées dans la littérature. Les approches d'interaction traditionnelles sont basées généralement sur le clavier et la souris. D'autres techniques d'interaction permettent aux utilisateurs d'interagir d'une façon naturelle en utilisant leurs capacités sensorielles [BVB, 04]. Dans cette section, nous présentons les techniques d'interaction naturelles utilisées dans la RA.

5.1. Interaction gestuelle

Plusieurs travaux réalisés démontrent l'utilisation des gestes comme un moyen de communication dans différents domaines tels que la RA, la RV, la reconnaissance de la langue des signes et les interactions Homme-Machine [BER, 14].

Un geste est un mouvement corporel de la tête, des mains ou d'une autre partie de corps [5]. Pour Cadoz, le geste est l'un des canaux de communications les plus riches. L'un des dispositifs les plus utilisés pour la reconnaissance des mouvements des mains et du corps est le kinect [FAB, 14].

Le geste de la main est une modalité très employée dans l'interaction naturelle. Ainsi, la main peut servir à manipuler des objets virtuels dans la RA. Par rapport à la richesse de l'information véhiculée par les gestes de la main, les possibilités de communication avec les ordinateurs sont aujourd'hui réduites avec la souris et le clavier [SIC, 08].

Parmi les travaux qui ont été exposés dans le contexte d'interaction basée sur les gestes de la main dans la RA, nous citons le travail de l'entreprise Ford de Bretagne, cette dernière a lancé

une campagne de RA pour promouvoir une voiture de 7 places (Grand C-MAX) sur des écrans. La campagne permet aux gens de gérer et d'explorer un modèle 3D virtuel de véhicule multi-personnes à l'écran grâce à la reconnaissance des gestes de leurs mains comme (figure 6) [3].

Figure 6 : *Interaction basée sur des gestes de la main pour une application de RA [3].*

5.2. Interaction basée sur le regard

Le contrôle par le regard est un principe sur lequel les laboratoires de recherche et les universités travaillent depuis quelques temps. Il est utilisé notamment pour faciliter le quotidien des handicapés [STJ, 15].

L'oculométrie (en anglais Eye-tracking ou Gaze-tracking) regroupe un ensemble de techniques permettant d'enregistrer les mouvements oculaires. L'oculométrie fait appel à ce que l'on appelle un oculomètre (en anlglais Eye-tracker). Ce dernier est un dispositif qui permet de suivre les mouvements de l'œil et de connaître la localisation du regard. La principale difficulté technique est de repérer de manière suffisamment fine la position et les mouvements de l'œil [BRM, 04].

L'oculométrie pourrait bien nos faire gravir une nouvelle marche dans la façon d'interagir avec nos écrans et leurs contenus numériques. Dans l'univers des jeux vidéos, Le Suèdois Tobii propose aux joueurs d'interagir via des solutions d'oculométrie dans le jeu Assassin's Creed : Rogue [STJ, 15].

L'interaction par le regard est une modalité naturelle très intéressante. Cette modalité a été expérimentée afin de réaliser des tâches de sélection et de manipulation comme illustrée dans la figure 7.

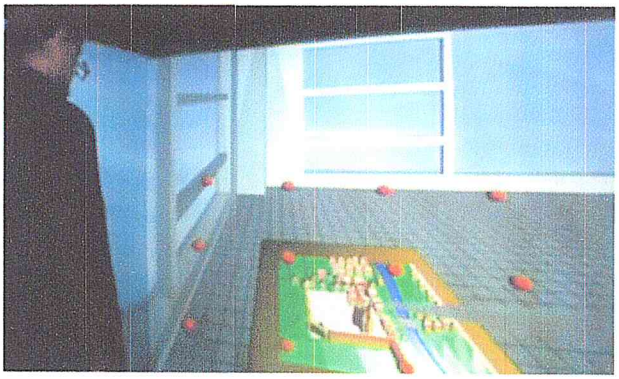


Figure 7 : *Sélection par le regard dans un environnement virtuel [PFE, 08].*

5.3. Interaction à base d'émotions

Le visage joue un rôle prépondérant en langue des signes, notamment par le sens porté par ses expressions. Peu d'études existent sur les expressions faciales en langue des signes, cela est dû au manque d'outils de description.

L'analyse du visage est une discipline dont les premiers travaux sont liés à l'essor de l'intelligence artificielle (IA). Les principales taches de l'analyse de visage sont [HME, 07] : la détection de visage, la reconnaissance de visage, la reconnaissance d'émotions et la description des mouvements faciaux.

La reconnaissance des visages en réalité augmentée permet de superposer à l'écran de l'appareil (un smartphone par exemple) d'un utilisateur, des informations relatives à son identité numérique. La société Polar Rosea a notamment développé un outil de reconnaissance des visages en réalité augmentée (figure 8). En pointant votre téléphone sur le visage d'une personne inscrite au servisse, vous aurez la possibilité d'accéder a ses différents profils (Facebook, Skype ou encore Yahoo) enligne [VIR, 16].

Figure 8 : *Reconnaissance* des visages en réalité augmentée *[VIR, 16].*

5.4. Interaction par l'activité cérébrale

Une interface cerveau-ordinateur appelée ICO (en anglais Brain Computer interface : BCI) ou interface neuronale directe (IND), permet aux utilisateurs d'interagir avec un ordinateur seulement par leurs signaux biologiques du cerveau sans la nécessité d'utiliser les muscles. Les ICOs sont utilisées par les handicapés et aussi par les personnes en bonne santé. ICO est un domaine de recherche émergent, mais il est encore relativement immature [ASL, 13].

L'envoi de commandes par l'activité cérébrale est un autre moyen d'interaction naturelle en RA. Une équipe de chercheurs réunis au sein de Joint Robotics Laboratory (JRL) a développé une interface cerveau-ordinateur qui permet un contrôle d'un robot (figure 9). Une personne munie d'un casque EEG se tient devant un écran d'ordinateur, relié au rebot qui est équipé de camera afin de fournir une vue subjective. Grace à un procédé de RA des flèches qui clignotent très rapidement se superposent à l'image réelle. La personne fixe son regard sur l'une des flèches pour mouvoir le rebot [ZEF, 12].

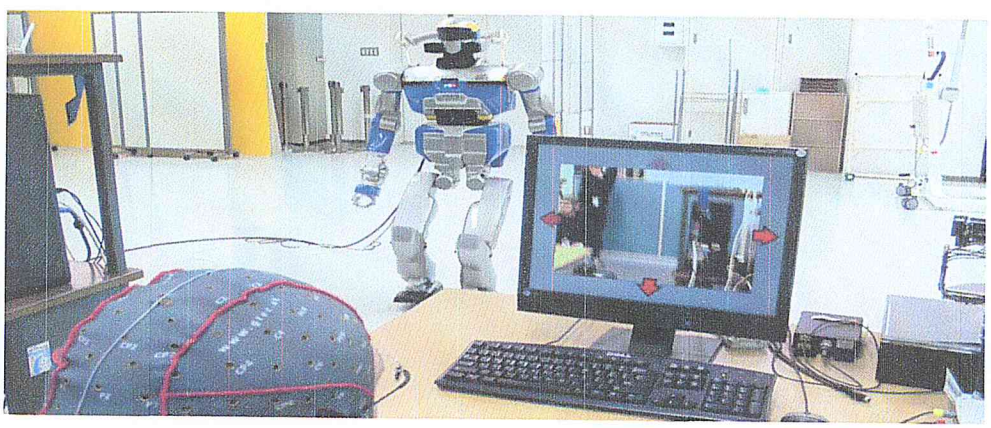


Figure 9 : *Contrôle de robot par la pensée [ZEF, 12].*

5.5. Interaction vocale

La voix a été exploitée comme moyen d'interaction dans plusieurs applications de RA. En effet, en plus d'être un moyen d'interaction naturel et familier, la modalité vocale offre à l'utilisateur la possibilité de sélectionner et de manipuler des objets 3D.

L'interaction vocale a été le centre de nombreuses études diverses et variées. En effet, de nos jours, elle fait l'objet d'une demande importante afin qu'on puisse tous communiquer intégralement de manière naturelle avec les machines.

Étant donné que la parole libère la vue et les mouvements, elle est aisément utilisée en superposition à d'autres modes de communication [MIN, 02]. Le module de commande vocale fait appel à la reconnaissance de la parole, La figure 10 montre les différentes étapes d'une approche globale du processus de reconnaissance de la commande.

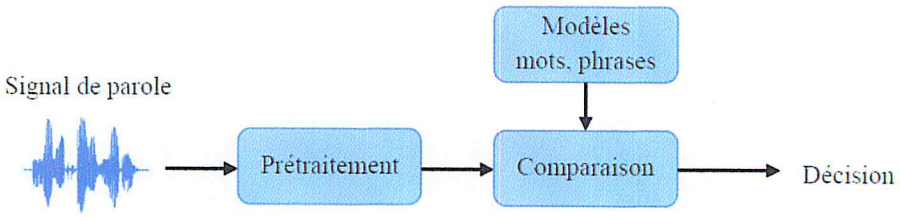


Figure 10 : *Schéma global d'un système de reconnaissance vocale.*

L'approche globale consiste à rechercher, pour chaque mot, la plus grande ressemblance avec l'un des modèles mémorisés.

Il est possible d'identifier plusieurs domaines d'application dans lesquels, notamment plusieurs tâches doivent être effectuées en parallèle dans des domaines divers : industriel avec les systèmes de commande (manipulation d'objets, d'automates, etc.) ou de saisie de données (lors du contrôle de processus, d'observations microscopiques, d'inspection de matériels, par exemple).

Le projet RASA [MCW, 00] propose un environnement de RA 2D multimodale par utilisant un système de reconnaissance textuelle et un système vocale.

Les Google Glass (présentées dans la figure 11) sont des lunettes à RA issues du projet Glass de Google, Elle permet d'accéder à la plupart des fonctionnalités de Google : agenda, SMS/MMS,

GPS, appareil photo, email, etc. L'interaction avec les Google Glass s'effectue par la dictée vocale [4].

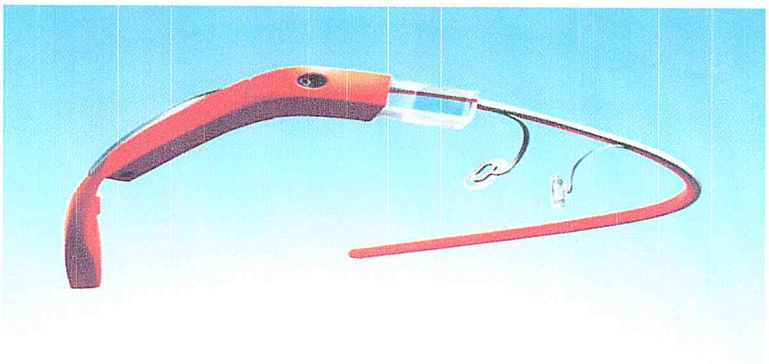


Figure 11 : *Google Glass (lunettes à réalité augmentée) [4].*

5.6. Interaction multimodale

La multi modalité reflète le caractère de multiplicité (préfixe multi) des modalités pour un même système interactif. Un système est multimodal s'il dispose d'au moins deux modalités pour un sens donné (entrée ou sortie). Ainsi, le système peut être qualifié de multimodal en entrée (ou en sortie) si au moins deux modalités d'entrée (ou de sortie) sont disponibles [JBO, 06].

L'interaction multimodale est basée sur la fusion entre plusieurs modalités, comme par exemple la voix, le geste [HAY, 11]. Dans [MCW, 00] les auteurs présentent un système de réalité augmentée dit le Rasa, où ils proposent une interaction basée sur la fusion entre la reconnaissance des gestes, les reconnaissances de la parole et la reconnaissance de texte et de symboles.

6. Classification des différentes techniques d'interaction

En se basant sur les travaux réalisés dans [MAA, 12; COA, 97; DIA, 08; HAA, 04; OHA, 11], une comparaison entre les différentes types d'interaction a été effectuée selon deux aspects : le respect des différentes caractéristiques identifiées (caractéristiques communes aux interfaces naturelles) et la capacité de réaliser les quatre tâches d'interaction universelles (navigation, sélection, manipulation et contrôle d'application).

Le tableau 1 montre la première classification basée sur les caractéristiques. Les signes « + » et « - » désigne si une modalité satisfait ou non un critère. La répétition du signe « + » indique le degré de satisfaction d'une modalité à un critère donné [DJE, 13].

			Sensibilité Aux corn-mandes naturelles	Facilité d'utilisation	Facilité d'apprentissage	Interaction direct	Transparence	Souplesse et flexibilité
Main	Avec contact	Tactile	++	++	++	++	+	+
		Tangible	+-	++	++	+	+	-
	Sans contact		++	+	+	++	++	+
Voix			++	+	+	++	++	+
Regard			++	++	++	++	++	+-
Mouvement du corps			++	+-	+	++	++	-
Activité cérébrale			++	+-	+	+-	+-	-
Mouvement des lèves			-	-	-	-	-	-
Expression de visage			-	-	-	-	-	-

Tableau 1 : *Classification des interactions selon l'axe des caractéristiques [DJE, 13].*

Le tableau 2 montre la seconde classification basée sur la capacité de réaliser les tâches d'interactions standards citées ci-dessus. Le signe « + » indique qu'une interface permet la réalisation de la tâche associée, et le signe « - » indique le contraire [DJE, 13].

			Navigation	Sélection	Manipulation	Contrôle d'application
Main	Avec contact	Tactile	+	+	+	+
		Tangible	+	+	+	+
	Sans contact		+	+	+	+
Voix			+	+	+	+
Regard			-	+	+	-
Mouvement du corps			+	+	+	+
Activité cérébrale			+	+	+	-
Mouvement des lèves			-	-	-	-
Expression de visage			-	-	-	-

Tableau 2 : *Classification des interactions selon l'axe des tâches d'interaction 3D* universelles *[DJE, 13].*

7. Conclusion

Dans ce chapitre, nous avons présenté les différentes techniques d'interaction existantes dans la RA. Une comparaison de ces différentes techniques a été présentée.

Dans le but de faciliter l'interaction entre l'utilisateur et les interfaces nous nous sommes intéressés pour l'interaction vocale qui représente un des moyens efficaces pour interagir avec l'environnement 3D d'autant plus qu'il permet à l'utilisateur de réaliser certaines tâches qui sont parfois difficiles avec les autres modalités d'interaction tels que le geste.

Chapitre III :

Conception du système

1. Introduction

L'objet de ce chapitre est de concevoir notre Système de Commande Vocale en RA (SCVRA). Pour cela, il est nécessaire de bien étudier les besoins de l'application, de déterminer les problèmes et les potentiels de la solution projetée en considérant les aspects conceptuels et techniques. La finalité de cette étude est de proposer une solution qui répond au travail demandé et aussi qui correspond aux besoins des utilisateurs.

Ainsi, nous avons décidé d'entamer la conception en adoptons le processus 2TUP étendu et le langage de modélisation unifié UML.

2. Processus de développement 2TUP étendu

Le choix de la démarche à adopter est très important afin de mieux faire face aux contraintes de développement des logiciels et de réduire les risques d'échecs.

Nous avons opté pour le processus 2TUP (Two Track Unifiend Process) étendu afin de distinguer *l'étude fonctionnelle, l'étude interactionnelle ainsi que l'étude technique.* Cette démarche nous permet de décomposer le travail pour essayer de diminuer le fossé entre l'expression du besoin et la conception.

Ainsi, l'idée majeure du 2TUP étendu est que toute évolution d'un système peut se décomposer et être traitée parallèlement, suivant un axe fonctionnel/interactionnel et un axe technique. Voir (cf. annexe C) pour plus de détails sur le processus 2TUP étendu et son déroulement.

3. Langage de modélisation UML

Pour concevoir notre système, nous avons choisi le langage unifié pour la modélisation UML (Unified Modeling Language). Ce dernier se définit comme « *un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, à spécifier et documenter des systèmes, à esquisser des architectures logicielles, à concevoir des solutions et à communiquer des points de vue* » [ROV, 07]. UML s'articule autour de treize types de diagrammes, Dans

notre travail, nous utiliserons seulement trois types de diagrammes : diagramme de cas d'utilisation, diagramme de classes et diagramme de séquences (cf. annexe C).

4. Domaine d'application

En terme applicatif, nous avons ciblé le domaine de l'assemblage assisté par ordinateur. Ainsi, le but général de notre application est de permettre à l'utilisateur de pouvoir opérer à l'assemblage de voiture. En effet, plusieurs domaines notamment en industrie ont besoins de simuler l'assemblage de pièces afin de visualiser par exemple un produit ou encore valider un certain nombre de caractéristiques dessus, le design automobile recours souvent à des opérations d'assemblage assistées par ordinateur afin d'adopter un modèle définitif.

Plusieurs applications en RV/RA sont dédiées à l'activité d'assemblage, dans notre cas nous effectuons un scénario d'assemblage de voiture dans un environnement de RA que nous développons. Pour la partie interactive, nous avons choisi d'utiliser pour l'implémentation de notre système une librairie de reconnaissance vocale basée sur un serveur dédié à ce faire, plus précisément, nous avons opté pour "UDP voice recognition Server" que nous intégrons à notre solution, le tout est développé sous le moteur graphique Unity 3D.

5. Conception du système de commande vocale en RA (SCVRA)

5.1. Etude préliminaire

L'étude préliminaire est la toute première étape du processus 2TUP étendu. Elle consiste à effectuer un premier repérage des besoins fonctionnels, d'interactions et techniques, en utilisant principalement du texte ou des diagrammes très simples. Elle prépare les activités plus formelles de capture des besoins fonctionnels/d'interaction et de capture de techniques [ROV, 07].

5.1.1. Présentation du projet à réaliser

Notre projet s'inscrit dans le cadre du projet de recherche intitulé « ***Interaction 3D collaborative et mobile dans un environnement de réalité virtuelle et augmentée*** » initié au sein de l'équipe Interaction Homme-Machine, Réalité Virtuelle et Augmentée (IRVA) au niveau du Centre de Développement des Technologies Avancées (CDTA).

Notre objectif consiste à concevoir puis réaliser un système de commande vocale basée sur une stratégie de reconnaissance vocale performante pour contrôler des objets 3D dans un environnement de RA. Le système proposé sera testé sur un exemple pratique.

5.1.2. Choix techniques

Du point de vue technique nous distinguons deux types de besoins : les besoins logiciels et les besoins matériels.

Les besoins logiciels identifiés sont :

- Programmation avec le langage C#.
- Utilisation du SGBD relationnel (MYSQL) et du langage SQL.
- Utilisation de moteur graphique Unity 3D.

Les besoins matériels recensés sont :

- Un microphone.
- Un PC portable.
- Une Webcam.
- Un marqueur pour la localisation des objets 3D.

5.1.3. Identification des acteurs

Les acteurs susceptibles d'interagir avec notre système sont (voir la figure 12) :

- Administrateur.
- Expert.
- Utilisateur Simple.

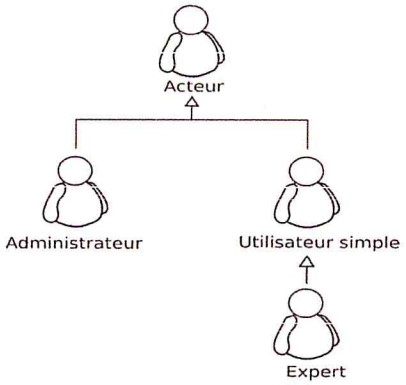


Figure 12 : *Acteurs impliqués dans le système.*

Le rôle de chaque acteur est spécifié dans le tableau 3 suivant :

Acteur	Rôle
Administrateur	• S'authentifier • Ajouter un compte d'utilisateur • Modifier un compte d'utilisateur • Supprimer un compte d'utilisateur
Utilisateur Simple	• S'authentifier • Afficher les composants 3D par famille • Sélectionner un composant 3D • Désélectionner un composant 3D • Manipuler un composant 3D / une voiture 3D (Rotation, Translation, Zoom) • Assembler les composants 3D • Modifier la couleur d'une voiture 3D
Expert	• S'authentifier • Afficher les composants 3D par famille • Sélectionner un composant 3D • Désélectionner un composant 3D • Manipuler un composant 3D / voiture 3D (Rotation, Translation, Zoom) • Assembler les composants 3D • Modifier la couleur d'une voiture 3D • Ajouter un composant 3D • Ajouter une couleur pour la voiture 3D • Supprimer un composant 3D • Modifier les caractéristiques d'un composant 3D

Tableau 3 : *Les acteurs de notre système et leurs rôles.*

5.2. Capture des besoins techniques

5.2.1. Diagramme global des cas d'utilisation techniques

La figure 13 décrit les cas d'utilisation techniques du système à concevoir.

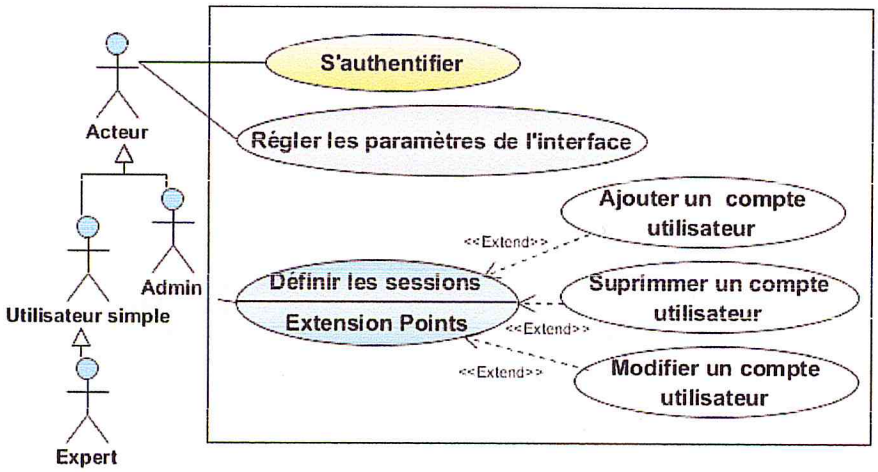


Figure 13 : *Diagramme global des cas d'utilisation techniques.*

5.3 Capture des besoins fonctionnels et des besoins d'interaction

5.3.1. Diagramme global des cas d'utilisation fonctionnels

La figure 14, défini le diagramme global des cas d'utilisation fonctionnels de notre système.

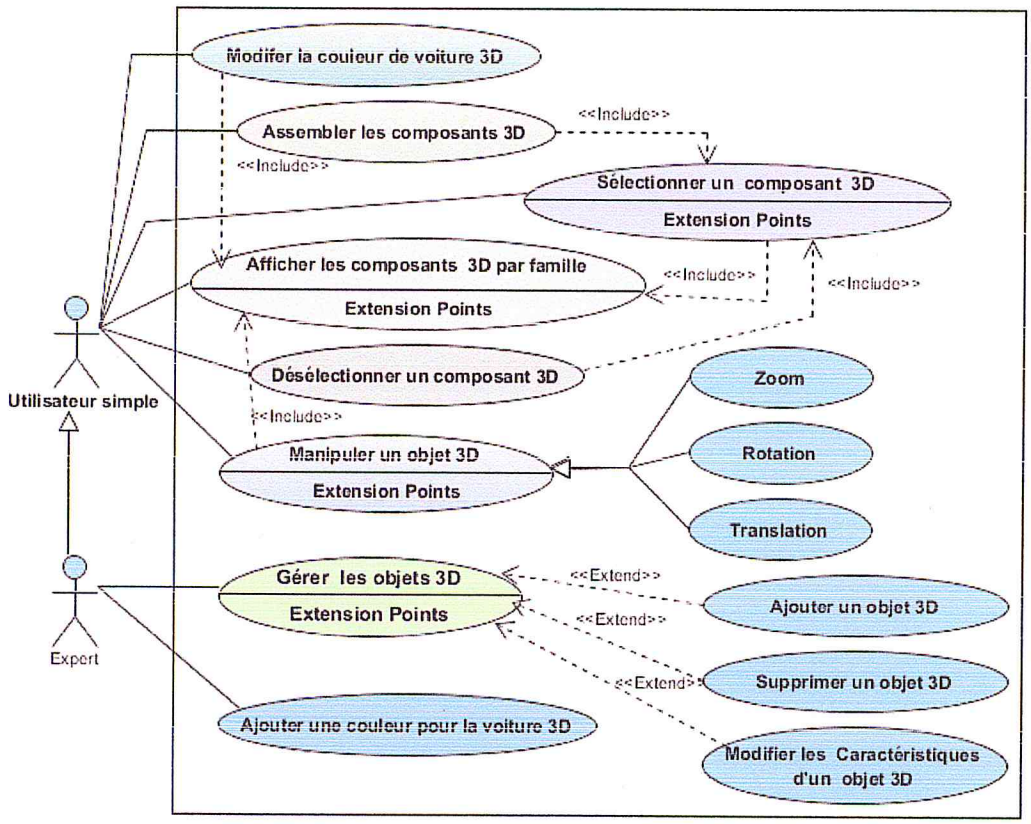


Figure 14 : *Diagramme global des cas d'utilisation fonctionnels.*

5.4. Analyse

Dans cette section, nous passons de la structure fonctionnelle à la structure composant. Nous distinguons deux types d'étude réalisée pendant cette phase : l'analyse statique et l'analyse dynamique.

- L'analyse statique qui décrit l'organisation des espaces métier et d'interaction sous forme d'attributs et de méthodes. Nous utilisons pour cela le diagramme de classes.
- L'analyse dynamique qui décrit les relations entre les différents composants des deux espaces. Nous utilisons pour cela le diagramme de séquences.

5.4.1. Développement du modèle statique

L'identification des cas d'utilisation, nous a permis d'identifier les classes candidates de notre système interactif, tout au long de notre développement.

La figure 15 représente les différentes classes de notre système :

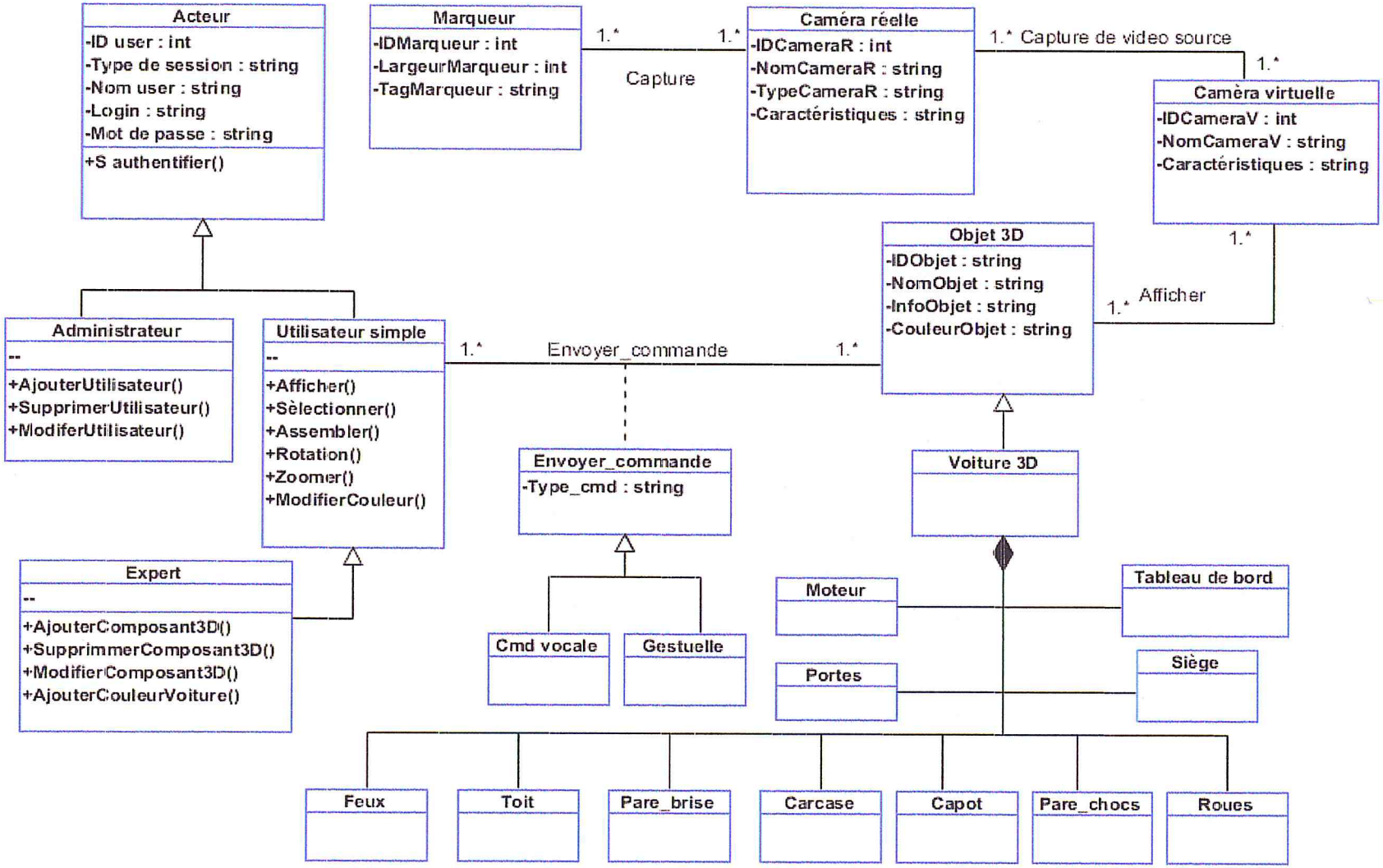


Figure 15 : *Diagramme de classe global de notre système.*

5.4.2. Développement du modèle dynamique :

Pour l'espace métier, l'analyse dynamique consiste à transformer les cas d'utilisation établis au cours des spécifications en diagrammes de séquences UML. Ces diagrammes représentent les liens possibles entre les instances des classes ainsi que les messages qui peuvent être échangés entre ces instances.

A Diagramme de séquence : manipuler un composant 3D

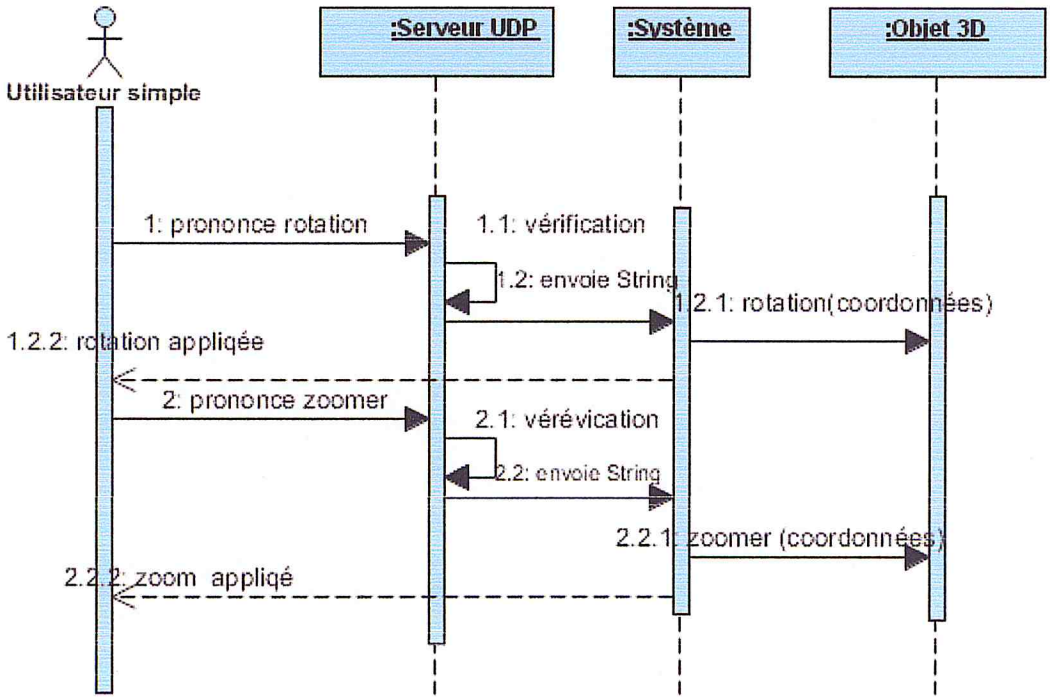


Figure 16 : *Diagramme de séquence : manipuler les objets 3D.*

Le scénario :

- **1** : L'utilisateur prononce une commande vocale pour manipuler (rotation) les objets 3D.
- **1.1** : Le serveur capture et traduit la commande vocale et la vérifie dans un fichier de grammaire.
- **1.2** : Le serveur envoie une commande vers le système pour manipuler (rotation) les objets 3D.

- **1.2.1 :** Le système manipule (rotation) les objets 3D.
- **2 :** L'utilisateur prononce une commande vocale pour manipuler (rotation) les objets 3D
- **2.1 :** Le serveur capture et traduit la commande vocale et la vérifie dans un fichier de grammaire.
- **2.2** : Le serveur envoie une commande vers le système pour zoomer les objets 3D.
- **2.2.1 :** Le système applique zoom sur les objets 3D.

B. Diagramme de séquence modifier la couleur de voiture 3D

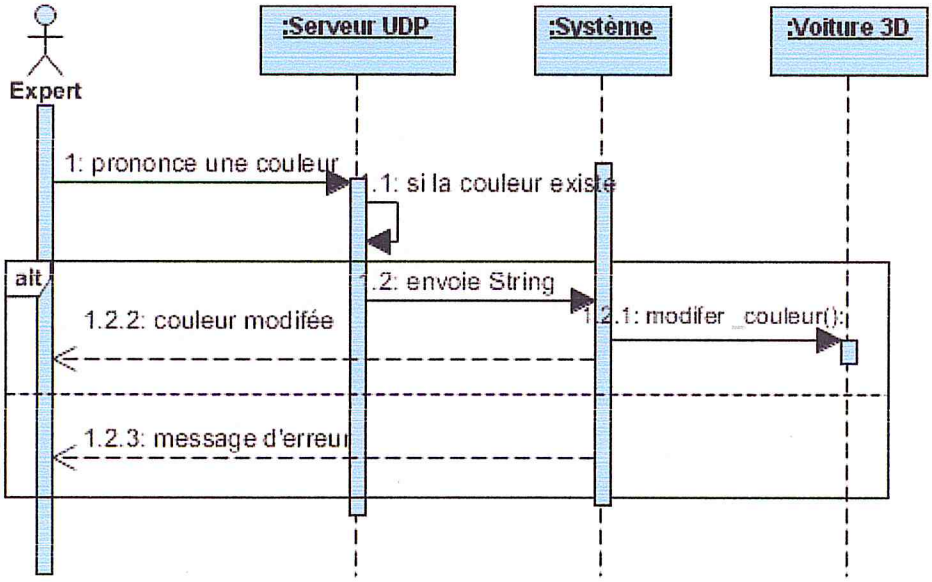


Figure 17 : *diagramme de séquence modifié la couleur de voiture 3D.*

Le scénario :

- **1** :L'utilisateur prononce une commande vocale pour modifier la couleur de la voiture
- **1.1 :** Le serveur qui va capturer et traduire la commande et vérifie dans un fichier de grammaire.
- **1.2**: Le serveur envoie une commande vers le système pour modifier la couleur de la voiture 3D.
- **1.6 :** Le système modifie la couleur de la voiture 3D.

C. Diagramme de séquence les opérations sur l'objet 3D.3D

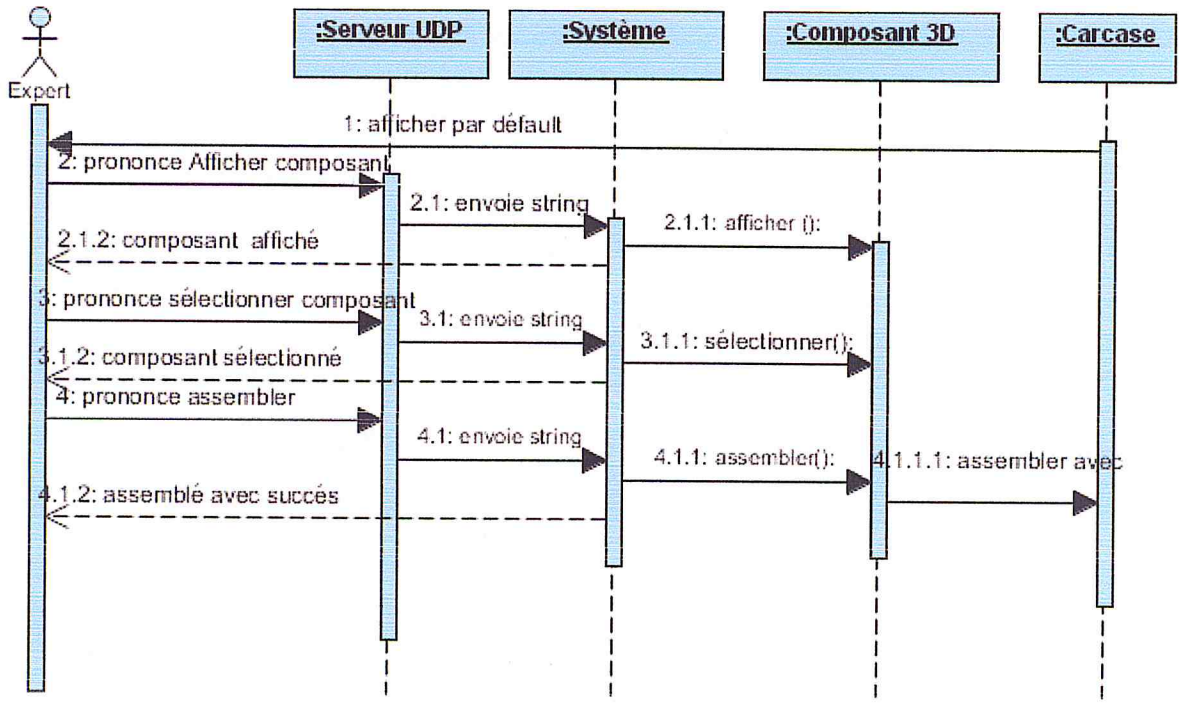


Figure 18 : *Diagramme de séquence les opérations sur l'objet 3D.*

Le scénario :

- **1 :** Le système affiche la carcasse par défaut.
- **2 :** L'utilisateur prononce une commande vocale pour afficher les composants 3D.
- **2.1:** Le serveur qui va capturer et traduire la commande et vérifie dans un fichier de grammaire et envoie une commande vers le système pour afficher les composants 3D.
- **2.1.1:** Le système affiché les composants 3D.
- **3 :** L'utilisateur prononce une commande vocale pour sélectionner un composant 3D.
- **3.1 :** Le serveur qui va capturer et traduire la commande et vérifie dans un fichier de grammaire et envoie une commande vers le système pour sélectionner un composant 3D
- **3.1.1 :** Le système sélectionné le composants3D.
- **4 :** L'utilisateur prononce une commande vocale pour assembler un composant 3D.
- **4.1 :** Le serveur qui va capturer et traduire la commande et vérifie dans un fichier de grammaire et envoie une commande vers le système pour assembler les composants 3D avec la carcasse.

➔ **4.1.1:** Le système assembler les composants 3D.

5.5. Conception détaillée

5.5.1. Description détaille des classes

Nous allons dans ce que suit décrire toutes les classes objets et association ainsi que leurs attributs :

Classe / Description	Attributs	Description	Type d'attribut
Utilisateur Classe qui regroupe tous les utilisateurs inscrits dans le Système	ID_user	Identifiant de l'utilisateur	Entier
	Nom_user	Nom et prénom de l'utilisateur	Chaine_C
	Login	Login de l'utilisateur	Chaine_C
	Mot_Passe	Mot de passe de l'utilisateur	Chaine_C
	Type_sesison	Type de sissone de l'utilisateur	Chaine_C
Administrateur Classe qui hérite de la classe (**Utilisateur**), Elle contient l'administrateur du système			
Utilisateur simple Classe qui hérite de la classe (**Utilisateur**), elle regroupe tous les utilisateurs simples du système			
Expert Classe qui hérite de la classe (**Utilisateur simple**), elle regroupe tous les experts du système			
Marqueur Classe qui définie le marqueur utilisé pour afficher le composant 3D	IDMarqueur	Identifiant du marqueur	Entier
	TagMarqueur	Nom du marqueur	Chaine_C
	LargeurMarqueur	Taille du marqueur	Réel
Composant 3D Classe qui regroupe tous les composants 3D	IDComposant	Identifiant de le composant 3D	Entier
	NomComposant	Nom de le composant 3D	Chaine_C
	InfComposant	Les informations sur le composant	Chaine_C

Voiture 3D Classe qui hérite de la classe (**Composant 3D**), elle regroupe les composants 3D			
Porte Classe qui hérite de la classe de voiture 3D			
Carcase Classe qui hérite de la classe de voiture 3D			
Roues Classe qui hérite de la classe de voiture 3D			
Tableau de bord Classe qui hérite de la classe de voiture 3D			
Moteur Classe qui hérite de la classe de voiture 3D			
Parabrise Classe qui hérite de la classe de voiture 3D			
Toit Classe qui hérite de la classe de voiture 3D			
Capot Classe qui hérite de la classe de voiture 3D			
Feux Classe qui hérite de la classe de voiture 3D			
Sieges Classe qui hérite de la classe de voiture 3D			
Caméra réelle Classe qui définir la caméra utilisée pour capturée le marqueur	IdCameraR	Identifiant de camera réel	Entier
	NomCameraR	Nom de la caméra réel	Chaine_C
	TypeCameraR	Type de la came	
	CaractéristiquesR	Caractéristiques pour la caméra	Chaine_C
Caméra Virtuelle Classe qui définir la caméra virtuelle utilisée pour capturer le vidéo source de	IdCameraV	Identifiant de caméra	Entier
	NomCameraV	Nom de la camera Virtual	Chaine_C
	CaractéristiquesV	Caractéristiques pour la caméra	Chaine_C

caméra réelle			
Envoyer _Commande Classe associative qui regroupe les deux méthodes qui utiliser	Type_cmd	La méthode de la commande	Chaine_C
Cmd vocale Classe qui hérite de la classe (**Envoyer _Commande**) Classe qui définir une méthode vocale			
Gestuelle Classe qui hérite de la classe de **(Envoyer _Commande)** Classe qui définir une méthode manuelle			

Tableau 4 : *La description des classes.*

5.5.2. Description détaillée des associations

Association	Classe		Attributs
Capteur Association qui spécifie la caméra qui capture le marqueur	**Camera réelle**	**1.***	
	Marqueur	**1.***	
Capteur de vidéo source Association qui spécifie la caméra virtuelle qui capture la vidéo source	**Caméra virtuelle**	**1.***	
	Caméra réelle	**1.***	
Afficher Association qui affiche le objet 3D	**Caméra virtuelle**	**1.***	
	Objet 3D	**1.***	
Envoyer_Commande Association qui spécifie la méthode utilisée	**Utilisateur simple**	**1.***	Type_cmd
	Objet 3D	**1.***	

Tableau 5 : *Description détaillée des associations.*

6. Conclusion

Dans ce chapitre, nous avons étudié les besoins de noter système et réalisé une conception tout en respectant leurs exigences. Grâce à la conception UML détaillée qui a été faite, nous pouvons maintenant passer à la réalisation du l'application.

Chapitre IV :

Implémentation du système et tests

1. Introduction

L'objectif de notre travail de master consiste à concevoir puis réaliser un système capable de reconnaitre des commandes vocales dans le but d'interagir avec des objets 3D dans un environnement de RA. Dans le chapitre précédent, nous avons présenté la conception de notre système, dans le présent chapitre, nous allons aborder la partie réalisation ainsi que le scénario sur lequel nous nous sommes basés pour la mise en œuvre et les tests de notre application. En effet, le système implémenté, devra reconnaitre des commandes simples et préalablement définies et les affectera par la suite à des actions au niveau de l'environnement 3D, ces actions reflètent des tâches d'interaction 3D.

Dans ce chapitre, nous présentons d'abord l'architecture globale de notre système. Nous détaillons également ses composants essentiels. Ensuite nous présentons nos tests et résultats. Enfin nous terminons ce chapitre par une brève conclusion.

2. Architecture globale du système SCVRA

Les trois acteurs de notre Système de Commande Vocale en RA, à savoir l'administrateur, l'expert et l'utilisateur simple disposent chacun d'une interface lui permettant d'interagir avec le système. L'administrateur est le responsable il ajoute, supprime et modifie les comptes des acteurs. L'utilisateur simple peut opérer à l'assemblage des différents composants de voiture 3D dans un environnement de RA. L'expert peut effectuer toutes les taches d'un utilisateur simple, il peut aussi ajouter/modifier/supprimer des composants 3D et ajouter des nouvelles couleurs pour la voiture 3D.

À l'exception de l'administrateur, les autres utilisateurs possèdent un marqueur, une caméra et un microphone, Une fois, la caméra capture le marqueur, un composant 3D (carcasse de voiture) sera superposé à l'image capturée, ainsi l'utilisateur peut afficher et contrôler les autres composants de voiture 3D, en prononçant des commandes simples et prédéfinies via le microphone.

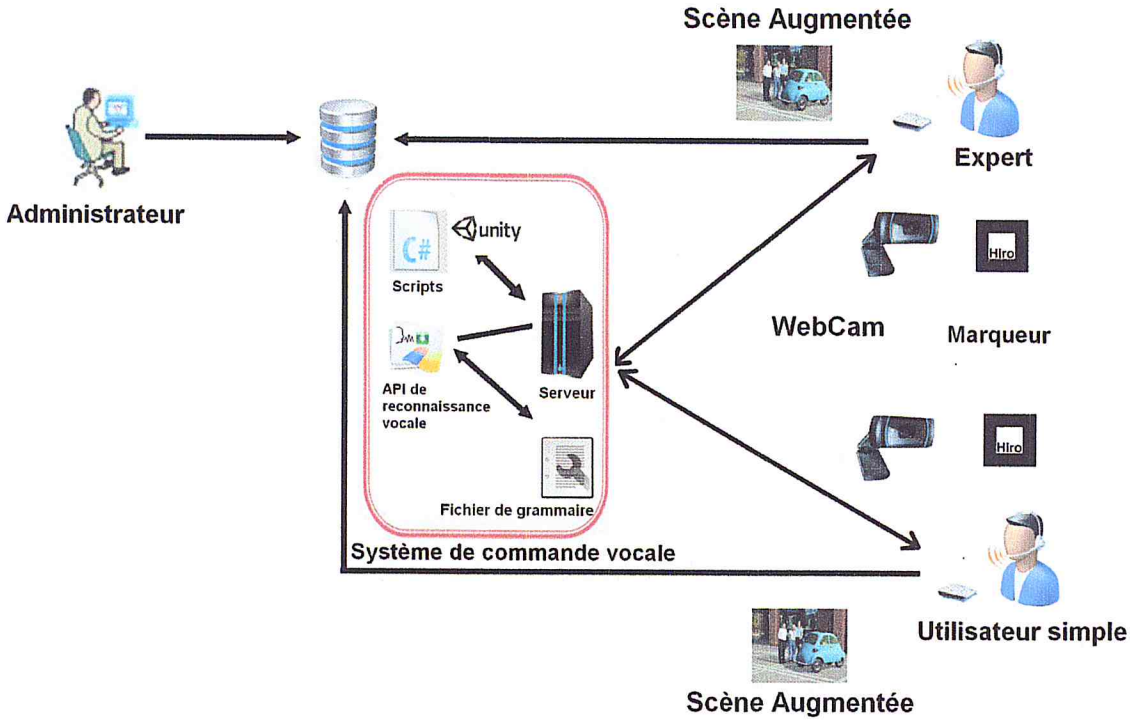


Figure 19 : *Architecture globale du système SCVRA.*

3. Pré-requis techniques

Pour la mise en œuvre de notre système, deux parties essentielles doivent être prises en considération :

3.1. Partie matérielle

Elle est constituée d'un ensemble de dispositifs destinés à la mise en marche de notre système. Il s'agit de :

- Pc portable ayant les caractéristiques suivantes :
  - Processeur Intel(R) Core(TM) i3-3217U 1.80 GHz.
  - Carte graphique AMD Radeon HD 7670M de 1Go.
  - 4 Go de Ram.
  - Système d'exploitation Windows 7 Edition intégrale 64 bits.
- Webcam utilisée pour capturer le flux vidéo de la scène réelle.
- Microphone utilisé pour l'acquisition du signal vocal.
- Un marqueur afin de mettre en correspondance les deux mondes : réel et virtuel.

3.2. Partie logicielle

Elle est constituée d'un ensemble d'outils logiciels permettant le développement et la mise en marche de notre système. Il s'agit de :

- Moteur de jeux Unity 3D.
- Le langage de programmation C#.
- Plugin ARToolKit pour Unity 3D.
- API de reconnaissance vocale.
- Le langage de programmation PHP.
- Le serveur MySQL.

3.2.1. Unity 3D

Unity 3D est une plateforme de création des jeux vidéo 3D, il s'agit d'un outil intégrant toutes les fonctionnalités nécessaires pour créer un jeu vidéo. Dans Unity 3D, le programme est piloté par des scripts qui peuvent être exprimés soit dans le langage C#, JavaScript ou Bodo.

Unity 3D dispose également d'une interface graphique riche et facilement personnalisable. Elle permet de travailler sur le jeu de manière interactive, elle permet par exemple de [GUE, 13] :

- placer les objets dans l'espace 3D.
- appliquer des transformations géométriques sur les objets de la scène à l'aide de la souris.
- modifier les textures, matériaux, propriétés physiques, paramètres de collision, etc.

L'interface graphique d'Unity 3D se décompose en cinq panneaux [DEV, 14] (figure 20) :

- Scène : interface pour éditer les éléments de notre scène.
- Jeu : prévisualisation de notre travail.
- Hiérarchie : recensement de tous les objets composants notre scène.
- Projet : différents éléments ajoutés dans notre projet.
- Inspecteur : panneau pour modifier les propriétés de l'objet du jeu sélectionné.

Figure 20 : *Interface graphique de moteur de jeu Unity 3D version 5.2.2f1.*

3.2.2. Langage C#

Le C# est un langage de programmation orienté objet créé par Microsoft en 2002. Il est Utilisé par un nombre important et grandissant de professionnels, il permet de réaliser des applications qui s'exécutent sur le Framework .NET de Windows. Le C# est un langage dont la syntaxe ressemble un peu au C++ ou au Java [HIL, 13].

Puisque nous utilisons le moteur de jeux Unity 3D pour l'implémentation de notre système, nous devons réaliser des scripts exprimés soit en C#, JavaScript ou Bodo [GUE, 13]. Dans notre cas, nous avons choisi le langage C# pour les raisons suivantes :

- La maitrise du langage : en effet, nous avons déjà une expérience de C# dans notre cursus.
- Langage de l'équipe accueillante : l'équipe IRVA travaille avec C# en RV/RA.
- Documentation riche pour les projets Unity 3D : une communauté anglophone très active.

Lorsque nous cliquons sur un script C#, par défaut Unity 3D utilise l'environnement de développement intégré (IDE) MonoDevelop, mais nous pouvons sélectionner un autre IDE par exemple Visual studio [UNI, 15]. Nous avons choisi MonoDevelop pour les raisons suivantes :

- MonoDevelop est l'IDE fourni avec Unity 3D, ce moteur de jeu s'intègre mieux avec Monodevelop.
- Environnement de développement complet et simple d'utilisation sans avoir besoin d'installer Visual studio (espace disque couteux).

Un script C #, une fois crée se présentera comme une classe dérivée d'une classe de base appelée MonoBahavior et portant le nom de fichier crée. Par défaut, cette classe possède deux méthodes de base, la méthode Start () et la méthode Update () :

```
1 using UnityEngine;
2 using System.Collections;
3
4 public class Test : MonoBehaviour {
5
6     // Use this for initialization
7     void Start () {
8
9     }
10
11    // Update is called once per frame
12    void Update () {
13
14    }
15 }
```

Figure 21 : *Contenu initial d'un script C#.*

- Le code qui gère le déclenchement des événements et les mise à jours des composants doit être placé dans la méthode Update ().
- Le code qui permet d'initialiser le programme doit être placé dans la méthode Start () [UNI, 15].

3.2.3. Plugin ARToolKit pour Unity 3D

Dans le but d'intégrer la bibliothèque ARToolKit (cf. annexe B) à notre application implémentée sous le moteur de jeu Unity 3D, nous avons utilisé le plugin ARToolKit pour Unity 3D, ce dernier est distribué comme un package qui contient les scripts et les ressources nécessaires pour faire cette intégration [ART, 15]. Son utilisation est très intéressant car il décharge le développeur des étapes d'estimation de pose (localisation) et de suivi de caméra mais nécessite une compréhension approfondie de ses objets de configurations afin de les employer avec exactitude et dans le bon sens.

3.2.4. API de reconnaissance vocale

Pour implémenter la reconnaissance vocale sous le moteur de jeu Unity 3D, nous avons utilisé un serveur (UDP voice recognition server) qui est basé sur Speech API (SAPI), l'API de

reconnaissance vocale native de Windows. Nous allons détailler la méthode que nous avons utilisée pour mettre en œuvre la reconnaissance vocale sous Unity 3D dans la section 5.

3.2.5 Serveur MySQL

MySQL est un système de gestion de base de données (SGBD) qui utilise le langage SQL. Sa licence peut être libre ou propriétaire selon le type d'application. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, en concurrence avec Oracle ou Microsoft SQL Server. Sa popularité est due en grande partie au fait qu'il s'agit d'un logiciel Open Source et gratuit [LUS, 07]. Le rôle de MySQL est la gestion de notre base de données (stockage, récupération, modification, suppression des données) qui contient les informations concernant les utilisateurs de notre système.

3.2.6. Langage PHP

PHP (officiellement, ce sigle est un acronyme récursif pour PHP : HyperText Preprocessor) est un langage de scripts généraliste et Open Source, spécialement conçu pour le développement d'applications web [PHP, 01]. Dans notre système, nous avons créé un espace d'administration ainsi qu'un espace d'authentification pour les utilisateurs du système, pour ce faire nous avons utilisé des scripts C# implémentés sous Unity 3D pour créer les interfaces correspondantes et envoyer des informations pertinentes aux scripts PHP sur notre serveur. PHP va donc effectuer les actions demandées et communique avec MySQL et lui fait passer le travail, à son tour MySQL fait le travail que PHP lui a soumis. PHP renvoie au serveur que MySQL a bien fait ce qui lui était demandé. Dans notre cas, PHP est utilisé pour établir le lien entre MySQL et Unity 3D.

4. Composantes du système à mettre en œuvre et démarche adoptée

Afin de réaliser notre système, nous avons identifié deux composantes principales, à lesquelles nous ajoutons le module de gestion des utilisateurs, le tout est initié par un scénario d'assemblage puis guidé par un ensemble de commandes transcrites dans le fichier grammaire utilisé :

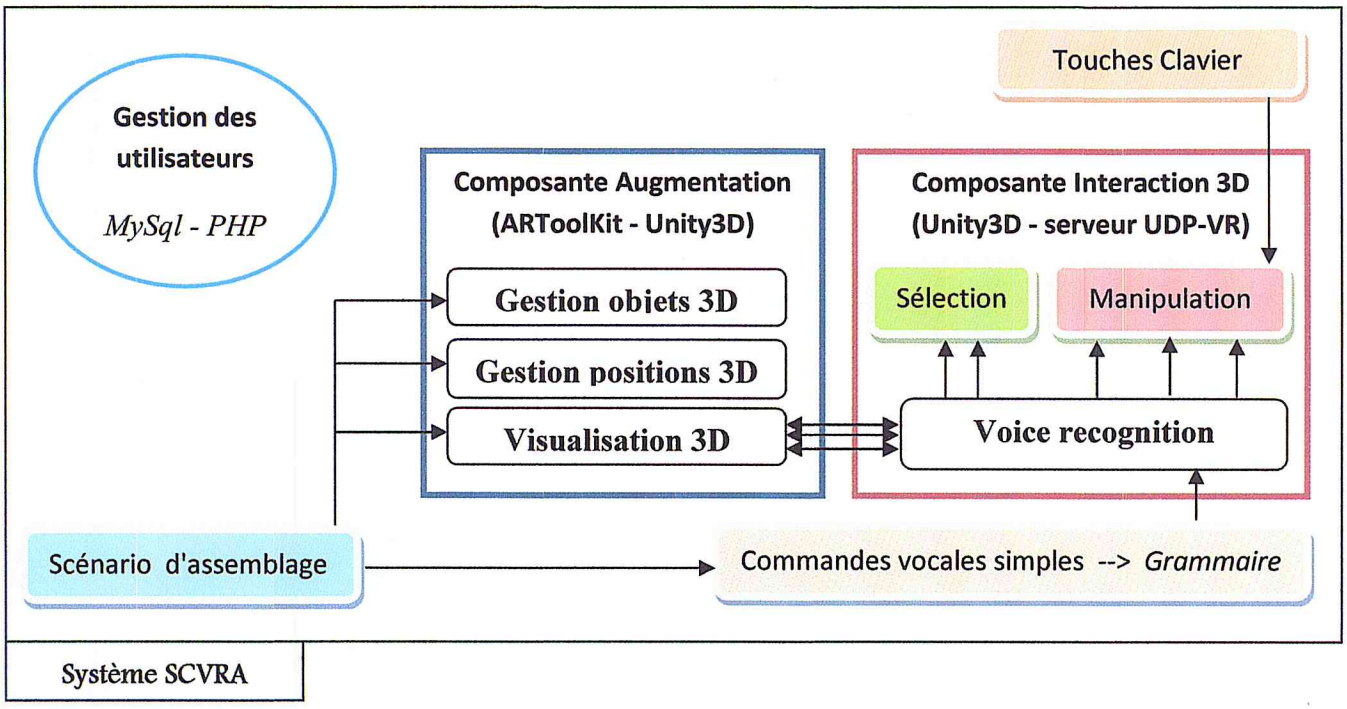


Figure 22 : *Composantes de SCVRA réalisé.*

4.1. L'Augmentation

L'augmentation est la première composante de notre système, il s'agit de pouvoir augmenter une scène réelle par l'ensemble des parties constituants un véhicule (objets virtuels) en vue de pouvoir exécuter des actions d'assemblage. En terme applicatif, nous avons adopté la démarche suivante :

4.1.1. Gestion des objets 3D

Afin qu'on puisse exécuter des actions d'assemblage de voiture, il nous fallait un modèle 3D de voiture, pour cela nous avons choisi un modèle 3D de voiture Peugeot 207 3 portes, notre choix a été basé sur le modèle qui contient les pièces essentielles pour la tâche d'assemblage selon le scénario de départ, aussi parce que ce modèle nous permet d'interagir avec chaque pièce individuellement, ce qui est très important pour pouvoir sélectionner chaque pièce et la manipuler à part entière.

La manière la plus simple pour importer un modèle 3D dans Unity 3D, est de faire glisser ce fichier dans la fenêtre du projet d'Unity 3D. Ce dernier peut lire juste les formats : .fbx, .dae (Collada), .3DS, .dxf et .obj. Une fois le modèle 3D de voiture Peugeot 207 3 portes importé,

nous avons rendu le design de ce dernier plus attrayant, nous avons changé les couleurs de ses composants (pièces), nous avons ajouté aussi des textures à ces composants en utilisant le panneau Inspecteur d'Unity 3D.

En terme de scénario, une fois la camera capture le marqueur, une carcasse de voiture Peugeot 207 3 portes est superposée à l'image capturée, positionnée, orientée et mise à l'échelle suivant les données fournies par le marqueur. Les autres composants de cette voiture sont organisés par famille (groupe) et n'apparaissent pas directement, cela nous permet de visualiser les pièces d'une manière simple et plus adéquate par rapport à l'espace d'affichage afin de ne pas encombrer la vue de l'utilisateur.

Chaque famille apparait à part, cela est géré par la seconde composante d'interaction et se fait par commande vocale. Le tableau suivant décrit l'ensemble des familles des composants de voiture (Peugeot 207 3 portes), ainsi que les objets virtuels correspondants que nous mettons à la disposition de l'utilisateur, les familles sont listées dans l'ordre de succession que nous avons retenu dans le scénario établi :

Nom de famille	Ordre d'apparition	Composants
Salon	1	Tableau
		Siège
Capot	2	Capot
Portes	3	Porte gauche
		Porte droite
		Porte arrière (coffre)
Pare-chocs	4	Pare-choc Avant
		Pare-choc arrière
Externe	5	Pare-brise
		Toit
Roues	6	Roue avant gauche
		Roue avant droite
		Roue arrière gauche
		Roue arrière droite

Tableau 6 : *Organisation des différents composants de voiture 207 3 portes en famille.*

4.1.2. Gestion des positions d'objets 3D

Dans cette section nous allons aborder l'étape de prétraitement qui représente une étape clé pour l'implémentation d'assemblage des composants de voiture et qui a pour objectif de sauvegarder les positions initiales d'objets 3D. Comme nous l'avons déjà mentionné

précédemment, l'objet virtuel "voiture" est constitué de plusieurs composants (des objets virtuels) que nous mettons à la disposition de l'utilisateur. Dans Unity 3D, la position d'un objet (GameObject) est déterminée par le composant « Transform » qui se trouve dans le panneau Inspecteur de l'objet. En effet chaque objet dans une scène à un composant « Transform », ce dernier est utilisé pour stocker et manipuler la position, la rotation et l'échelle de l'objet. Chaque « Transform » peut avoir un parent, ce qui nous a permis d'appliquer la position, la rotation et l'échelle hiérarchiquement à nos objets. Dans notre cas nous avons opté pour la méthode suivante pour la gestion des positions d'objets 3D :

D'abord, nous avons sauvegardé les positions et les rotations initiales de tous les composants de modèle 3D utilisé. Pour le faire, nous avons dupliqué tous les objets virtuels (composants de voiture) pour obtenir les positions exactes de ces objets. Une fois les objets dupliqués sont crées nous avons supprimé les composants responsables de rendu de chaque objet dupliqué, il s'agit de Mesh Filter et Mesh Renderer (Mesh Renderer prend la géométrie de Mesh Filter et le rend à la position définie par le composant Transform de l'objet) donc nous gardons que le composant « Transform » pour chaque objet dupliqué, ce qui rend les objets dupliqués vides.

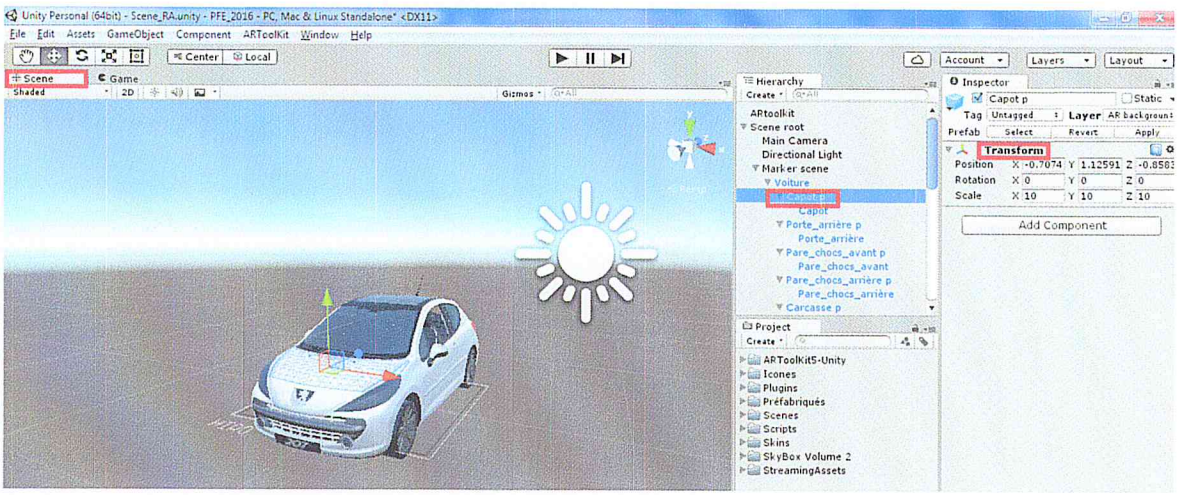


Figure 23 : *sauvegarder les positions initiales de tous les composants de modèle 3D.*

Ensuite, nous avons placé les composants de modèle 3D de voiture (objets virtuels) a des positions différents dans le monde virtuel (scène d'Unity 3D), la position de chaque objet dans l'espace du monde (world space) d'Unity 3D est représentée par un Vecteur3 (X, Y, Z), ce dernier représente la structure utilisée par Unity 3D pour passer les positions et les directions 3D. Nous avons aussi fait tourner (rotation) des objets virtuels autour des axes locaux (X, Y, Z) pour des fins des visualisations. Notons que tous les objets virtuels 3D de

voiture sont des fils d'un objet virtuel vide s'appelle voiture qui a son tour représente un fils d'un objet s'appelle ARMarker qui représente un marqueur de suivi dans le système (objet de configuration de RA).

Chaque objet qui à un parent dispose de deux positions, une position globale dans la scene et une position locale par rapport à son objet parent. La position locale est celle qui apparait dans le composant « Transform » dans le panneau Inspecteur d' Unity 3D si l'objet dispose d'un parent.

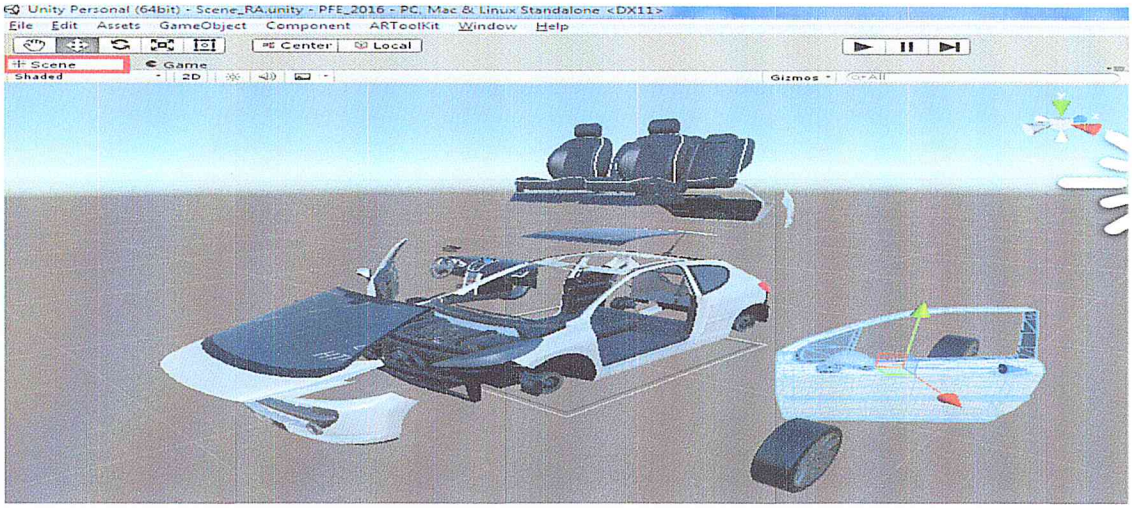


Figure 24 : *Placement des objets 3D a des positions différentes.*

Enfin, puisque les positions et les rotations initiales des objets virtuels sont sauvegardés dans des objets vides, nous avons crée un script qui permet a un objet de déplacer (animation) d'un point (position initiale) vers un point cible (position locale sauvegardé précédemment) dans une ligne droite suivant une vitesse précise. Au même temps ce script permet à l'objet de revenir à sa rotation initiale (déjà sauvegardée) via une vitesse déterminée.

4.1.3 .Gestion de la visualisation des objets :

Comme déjà mentionné, dès que la camera capture notre marqueur, la carcasse de modèle 3D (voiture Peugeot 207 3 portes) est superposée à l'image capturée. Pour le reste des composants de voiture, ils sont affichés par familles pour ne pas encombrer la vue de l'utilisateur. En effet nous avons décidé de travailler dans un environnement entièrement 3D, l'affichage de tous les composants de voiture (au total 14 composants) dans la scène va

charger tout l'espace d'affichage et donc nous risquons de ne pas visualiser les actions d'assemblage d'une manière convenable.

Par défaut, à l'exception de la carcasse de voiture, tous les autres composants (organisés par familles) sont cachés, pour ce faire, nous avons utilisé la propriété SetActive (false) qui rend un objet inactif, autrement dit elle désactive tous les composants d'objet (scripts, renderers...etc.), donc tous les scripts glissés sur un objet ne seront plus appelés. Ce module est très en relation avec la composante d'interaction car chaque visualisation est actionner par une commande vocale.

4.1.4. Implémentation de l'augmentation

Dans cette section nous allons présenter la méthode utilisée pour implémenter l'augmentation dans Unity 3D via le plugin ARToolKit pour Unity 3D :

Au niveau le plus élémentaire, ce que le plugin ARToolKit pour Unity 3D fait, est d'aligner une caméra virtuelle au sein d'Unity 3D avec une camera du monde réel (comme une webcam) par rapport à un marqueur de suivi. Par exemple, si nous pointons notre webcam sur un marqueur imprimé, la caméra virtuelle correspondante dans Unity 3D regarde à l'endroit équivalent dans le monde virtuel. Si nous plaçons alors un modèle 3D de voiture 207 3 portes dans le monde virtuel (Scène d'Unity 3D), et nous le superposons sur la vidéo entrante (Video source), nous produisons une vue en RA.

Après l'importation de package ARToolKit pour Unity 3D, notre projet Unity 3D contient donc les scripts et ressources nécessaires pour créer une scène de RA. Nous notons que les composants suivants travaillent ensemble pour créer une scène de RA :

- ARController : gère le fonctionnement global du plugin de suivi (initialisation générale, configuration, exécution et arrêt d'ARToolKit).
- AROrigin : représente le centre du monde ARToolKit et la racine de la scène.
- ARTrackedObject : représente le marqueur comme suivi dans l'espace, un contenu pertinent pour le marqueur sera attaché à ce parent.
- ARMarker : représente un marqueur suivi dans le système, nous devons ajouter un script ARMarker pour chaque marqueur individuel que nous souhaitons suivre.
- ARCamera : associe une caméra au contenu de RA.

Pour créer une scène de RA dans Unity 3D, nous avons suivi les étapes suivantes :

- Création d'un objet vide "ARController", ayant comme script ARController.
- Définition de script ARMarker pour l'objet vide "ARController" ayant comme Tag le marqueur choisi (nous avons utilisé un seul marqueur de base "Hiro" de type carré (square).
- Création d'un objet nommé "Scene root" pour définir le centre de la scène. Nous lui ajoutant le script AROrigin.
- Création d'une couche que nous avons nommé "AR Background" comme layer pour l'objet "Scene root", Il est également utile de mettre tous les enfants de l'objet "Scene root", dans sa propre couche "AR Background".
- Ajout d'un objet fils "Marker Scene" de l'objet parent "Scene root" et définition du script ARTrackedObject à cet objet avec le Tag défini plus haut.
- Ajout de l'objet Camera d'Unity 3D et lui attacher un script ARCamera. Cet objet doit être un fils de l'objet "Scene root".
- Ajout d'un objet fils voiture 3D de l'objet parent "Marker Scene" et réglage de l'échelle et la position de cet l'objet.
- Changement de la valeur de "Rotation: X" de l'objet "Scene root" à 90, pour que le marqueur repose à plat sur le sol de la scène virtuelle.

Cette première composante est très importante pour le développement de notre système, elle nous a permis d'augmenter une scène réelle par l'ensemble des pièces d'une voiture, de pouvoir opérer à certains prétraitements nécessaires pour disposer plus tard des positions adéquates lors de la phase d'assemblage. Même si l'augmentation est effectuée en un seul bloc, le module de visualisation nous a néanmoins permis de gérer l'affichage d'une manière intéressante à l'utilisateur final en permettant de visualiser les objets virtuels par famille. Cette partie nous a nécessité une compréhension technique d'Unity 3D, de ces composants, de ces scripts que nous avons employés pour atteindre notre objectif. L'ensemble des aspects techniques de cette partie est détailler dans (cf. annexe B).

4.2. L'interaction

L'interaction est la seconde composante de notre système, elle est basée sur des commandes vocales simples et prédéfinies. Nous détaillons ici la démarche adoptée permettant d'intégrer dans notre système un lecteur vocal en vue de contrôler des objets 3D pour une opération

d'assemblage de véhicule. Nous avons mis l'accent essentiellement sur deux tâches principales d'interaction 3D : la sélection ainsi que la manipulation et cela pour leur pertinence à l'application d'assemblage que nous développons, notamment dans un environnement de RA.

4.2.1. Commandes vocales pour l'interaction

Afin que l'utilisateur puisse interagir avec les objets 3D, nous mettons à sa disposition des commandes simples, ces commandes sont prédéfinies dans un fichier de grammaire. Le tableau suivant résume ces commandes vocales et la description de chaque commande, ainsi que son intérêt dans le monde 3D :

Commande Vocale	Description de la commande	Intérêt de la commande
Afficher (nom de famille ou composant)	Pour afficher les composants 3D par famille	Visualisation objet 3D
Sélectionner (nom de composant)	Pour sélectionner un composant	Interaction 3D - sélection
Désélectionner (nom de composant)	Pour désélectionner un composant	Interaction 3D - désélection
Assembler (nom de composant)	Pour assembler le composant sélectionné avec la carcasse	Interaction 3D manipulation (Translation + Rotation)
Rotation X/Y (45, 90, 180…)	Pour visualiser la voiture en différents angles	Interaction 3D manipulation Visualisation objet 3D
Position initiale	Pour visualiser la voiture en positon et rotation initiale.	Interaction 3D manipulation Visualisation objet 3D
Couleur	Pour modifier la couleur de la voiture	Visualisation objet 3D

Tableau 7 : *Description des commandes d'interaction.*

4.2.2. Interaction en utilisant les touches clavier

L'objectif de ce présent travail, est d'utiliser la voix pour interagir avec le monde 3D. Ceci dit, comme c'est le cas pour de nombreuses applications dans le domaine de l'interaction 3D, la voix est souvent utilisée comme une modalité combinée à d'autres techniques d'interaction (WIMP/post-WIMP), la raison principale est que certaines actions sont plus faciles par l'usage de la voix uniquement et d'autres sont par contre nettement meilleures par l'usage d'une autre technique d'interaction. Nous avons utilisé l'interface clavier pour l'implémentation de certaines fonctionnalités de système afin de faciliter la tâche à l'utilisateur par l'emploie de boutons, en terme applicatif les touches claviers sont utilisées juste pour la visualisation, certaines touches permettent d'avoir le choix d'opérer l'interaction soit par la voix ou le

clavier. Le tableau suivant résume les touches clavier et la description de leur usage, ainsi que son intérêt dans le monde 3D :

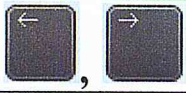
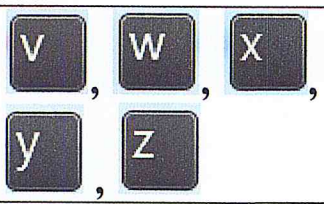
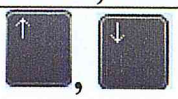
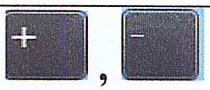
Touche clavier	Description	L'intérêt de la touche
	Visualiser la voiture en position et rotation initiale.	Interaction 3D manipulation Visualisation objet 3D
	Visualiser la voiture en différentes positions	Interaction 3D manipulation Visualisation objet 3D
	Visualiser la voiture en différents angles. Rotation selon l'axe x (90, -90, 180) successivement.	Interaction 3D manipulation Visualisation objet 3D
	Visualiser la voiture en différents angles. Rotation selon l'axe y (45, -45, 90, -90, 180) successivement.	Interaction 3D manipulation Visualisation objet 3D
	Visualiser la voiture en différents angles. Rotation autour l'axe y de 5 degrés	Interaction 3D manipulation Visualisation objet 3D
	Visualiser la voiture en tailles différentes (change d'échelle)	Visualisation objet 3D

Tableau 8 : *Description des touches clavier utilisées.*

4.2.3. Mise en œuvre de la reconnaissance vocale dans Unity 3D

Nos recherches concernant l'implémentation de la reconnaissance vocale dans Unity 3D, nous ont amenés au résultat qu'il n'existe pas une bibliothèque de reconnaissance vocale implémentée sous le moteur graphique Unity 3D. Dans ce sens, nous avons intégré la reconnaissance vocale par l'usage de l'API de reconnaissance vocale de windows (Speech API).

Le Framework .Net de windows contient une librairie de reconnaissance vocale System.Speech, qui a les caractéristiques suivantes [MCC, 14] :

- Fait partie du système d'exploitation (Windows Vista+).
- Ne peut pas être redistribué.
- Utilise des grammaires.
- permet la formation pour utilisateur spécifique.
- Code natif Speech API (C++).

Les versions actuelles d'Unity 3D ne supportent pas cette librairie, pour cette raison on a opté pour un serveur (UDP speech recognition server) qui fait l'intermédiaire entre Unity3D et L'API de reconnaissance vocale de Windows.

Les performances de notre système (SCVRA) dépondent en fait de l'apprentissage de système d'exploitation à la voix, en plus il faudra bien articuler les mots que nous prononçons. Le succès de la reconnaissance vocale est aussi lié à la qualité du microphone qu'on utilise, ainsi il faut bien positionner le micro lors de la prononciation des commandes, autrement dit :

- Placer le micro a 2 ou 3 cm de la bouche, sur un coté.
- Ne pas respirer directement dans le micro.

Nous avons mentionné que l'API de reconnaissance vocale de windows se base sur l'utilisation des grammaires, en effet notre fichier grammaire (grammar.txt) contient des commandes vocales simples que l'utilisateur prononce pour interagir avec les objets 3D. Parmi ces commandes nous trouvons dans le fichier grammar.txt (figure 25) :

grammar.txt

```
# commentaire
#I 127.0.0.1
#P 26000
#V 70
afficher salon
selectionner siege
deselectionner siege
selectionner tableau de bord
deselectionner tableau de bord
assembler
noir obsidien
blanc banquise
orange salamanque
.....
```

Figure 25 : *Contenu de notre fichier grammaire.*

Les premières lignes du fichier grammaire, permettent la configuration de certains paramètres.

>> Pour ajouter un commentaire.

#I >> L'IP du serveur. 127.0.0.1 car dans notre cas le serveur est sur la même machine que le client.

#P >> Port utilisé.

#V >> Validation de la reconnaissance. 0 à 100%. 70 est un bon chiffre.

4.2.4. Système mise en place pour le lecteur vocal

La figure 26 montre le schéma de système mise en place pour le lecteur vocal, nous présentons dans cette section, le fonctionnement de ce système.

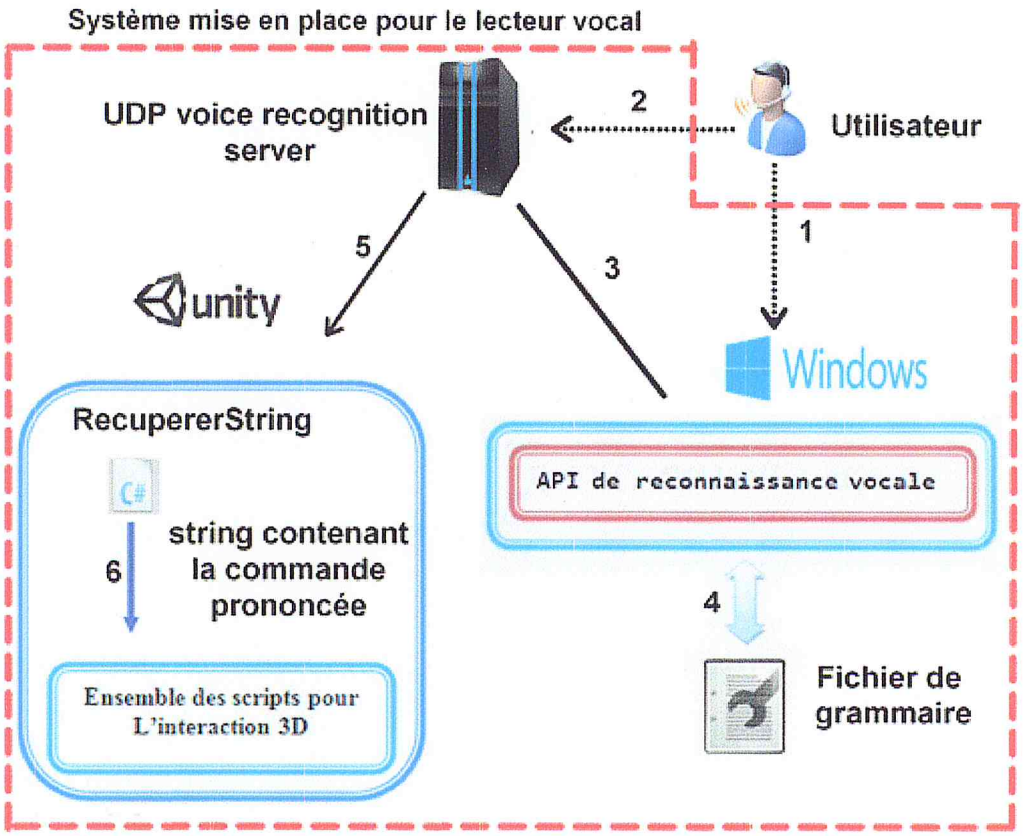


Figure 26 : Schéma de *systéme mise en place pour le lecteur vocal.*

1. Un nouveau utilisateur devra entraîner son ordinateur à reconnaître sa propre voix, en effet l'utilisation de l'API de reconnaissance vocale de Windows, nécessite la formation des utilisateurs (un utilisateur lit un texte et le système apprend à comprendre la prononciation de cet utilisateur particulier).

2. Après l'authentification au système SCVRA, un utilisateur (simple ou expert) prononce une commande vocale via le microphone.

3. Le serveur utilise L'API de reconnaissance vocale de windows pour la reconnaissance vocale de la commande prononcée.

4. Après l'analyse de la voix de l'utilisateur captée au moyen du microphone et sa transformation en chaine de caractère, l'API de reconnaissance vocale de windows compare cette chaine de caractère avec le contenu de fichier grammaire.

5. S'il y'a une correspondance entre la chaine de caractères avec l'une des lignes du fichier grammaire, Le serveur récupère la chaine de caractère contenant la commande prononcée et l'envoie via les sockets à Unity3D.

6. Un script C# implémenté sous Unity3D, nommé RecupererString, est utilisé pour récupérer la chaine de caractère depuis le serveur via des sockets UDP. Ensuite des scripts C# contiennent des instructions (Switch case) qui sont utilisés pour transformer cette chaine de caractère à des actions au niveau de l'environnement de RA.

Cette seconde composante est capitale pour permettre une interaction 3D avec les objets virtuels de la scène augmentée, comme tâche d'interaction nous avons notamment mis l'accent sur la sélection et la manipulation 3D (translation, rotation et mise à l'échelle) que nous effectuons dans le bon sens pour permettre des actions d'assemblage, notre travail exige l'usage d'une seule modalité d'interaction qui est la voix, nous avons trouvé judicieux d'ajouter une interaction via l'interface clavier dans le soucis d'enrichir les possibilités offertes à l'utilisateur, ainsi par l'usage de touches clavier nous permettons une meilleure visualisation du véhicule assemblé.

Cette partie a été réalisé notamment par le codage d'un ensemble de scripts que nous rattachant aux objets 3D sous Unity 3D, la reconnaissance vocale a été intégrée par l'usage du serveur UDP voice recognition et par l'emploi de L'API de reconnaissance vocale de windows d'une manière complètement transparente à l'utilisateur final.

5. Présentation de l'application

Dans ce qui suit, nous allons présenter quelques captures d'écran de notre système. Dans ce dernier, nous avons proposé plusieurs interfaces graphiques. La première fenêtre qui s'affiche au lancement de notre application est celle de l'authentification, l'utilisateur doit indiquer son pseudonyme et son mot de passe pour se connecter au système (figure 27).

Figure 27 : *Interface d'authentification.*

Les trois acteurs de notre système à savoir l'administrateur, l'expert et l'utilisateur simple disposent chacun d'une interface lui permettant d'interagir avec le système.

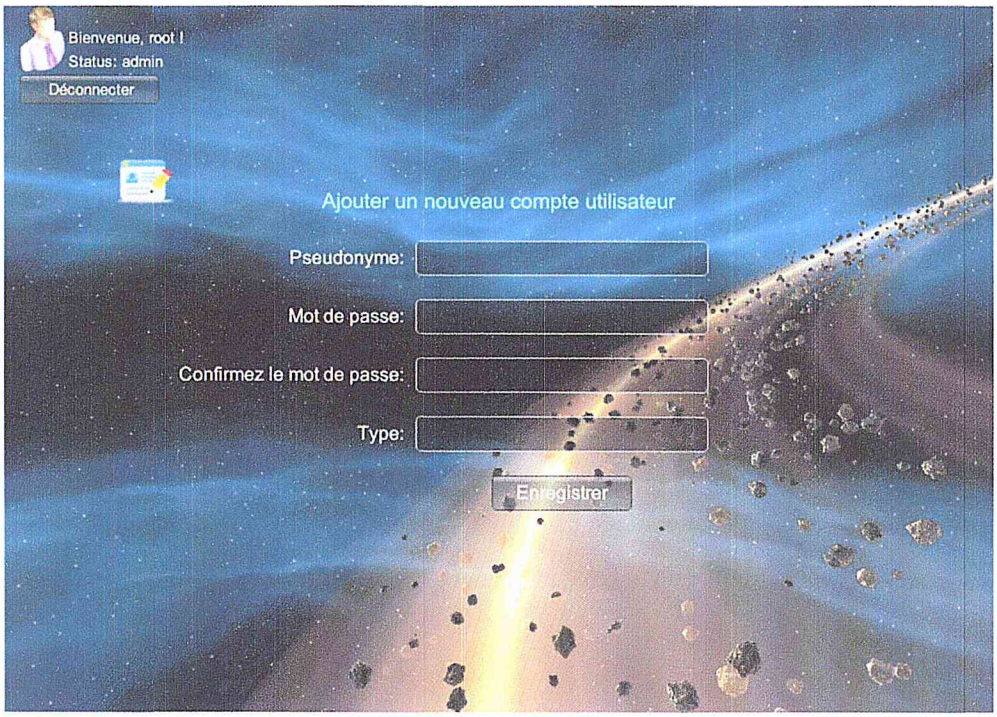


Figure 28 : *interface d'administration, ajout d'un nouveau compte utilisateur.*

Maintenant, nous allons présenter quelques captures d'écran qui montre quelques tests de notre application, en se basant sur le scénario prédéfini précédemment.

Une fois, la camera capture le marqueur, une carcasse de voiture (Peugeot 207) est superposée à l'image capturée (figure 29).

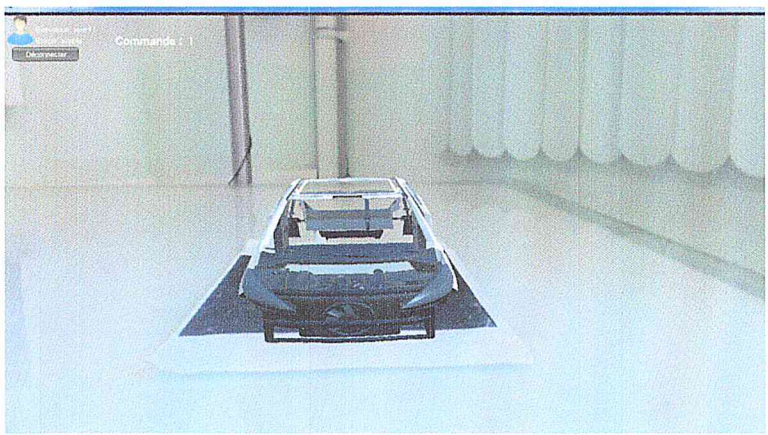


Figure 29 : *Superposition de carcasse de voiture à l'image capturée par la caméra.*

La Figure 30 montre l'apparition des composants de famille salon (siège, tableau) après qu'un utilisateur prononce la commande « afficher salon » via le microphone.

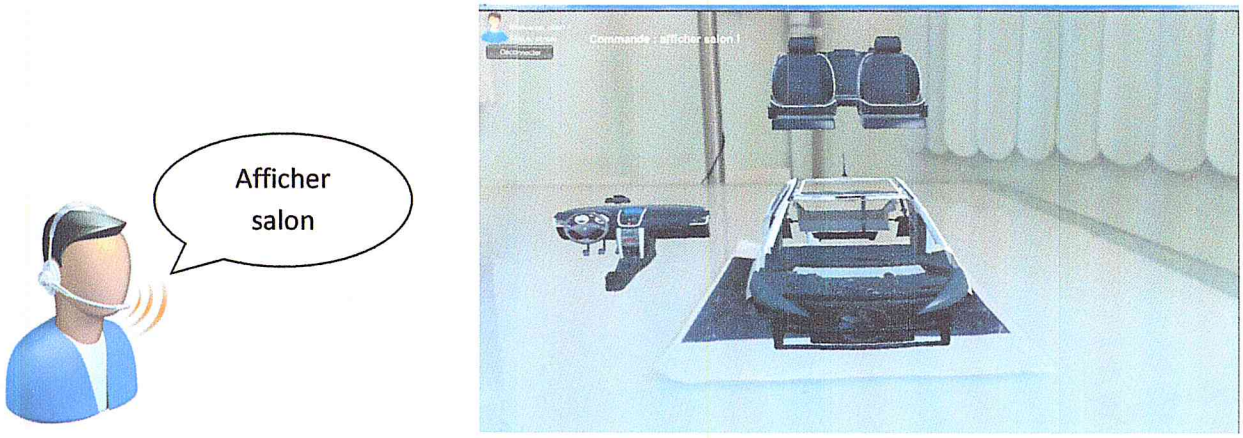


Figure 30 : *Apparition des composants 3D de famille salon (siège, tableau).*

Une fois les composants de famille salon apparaissent, l'utilisateur a le choix de sélectionner n'importe composant de cette famille afin qu'il puisse l'assembler. Les composants selectionnés sont colorés en vert. La figure 31 montre une sélection du composant siège.

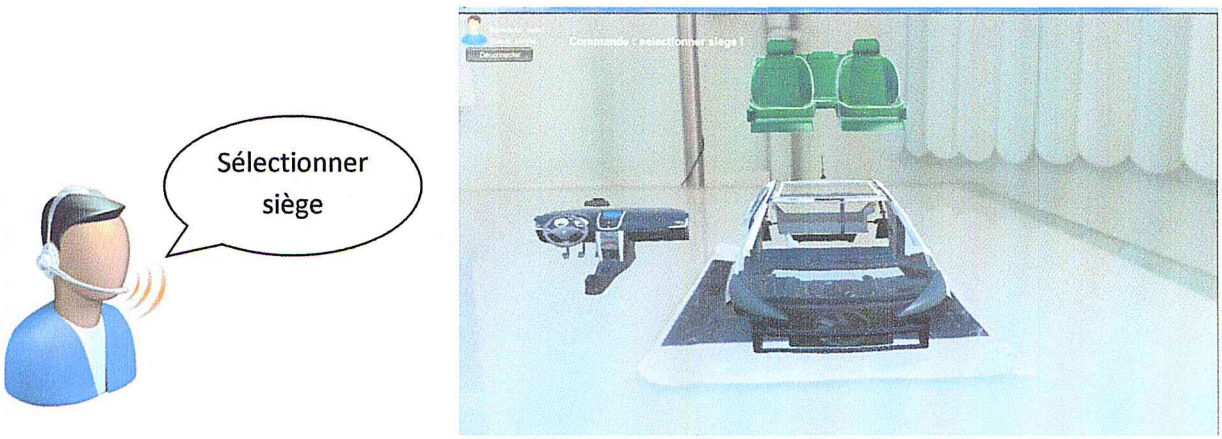


Figure 31 : *Sélection d'un composant 3D (siège).*

Lors de la prononciation de la commande « assembler », tous les composants sélectionnés seront colorés en rouge et chacun d'eux se déplacera à une position prédéfinie, les figure 32 et 33, montrent l'opération d'assemblage de composant 3D siège.

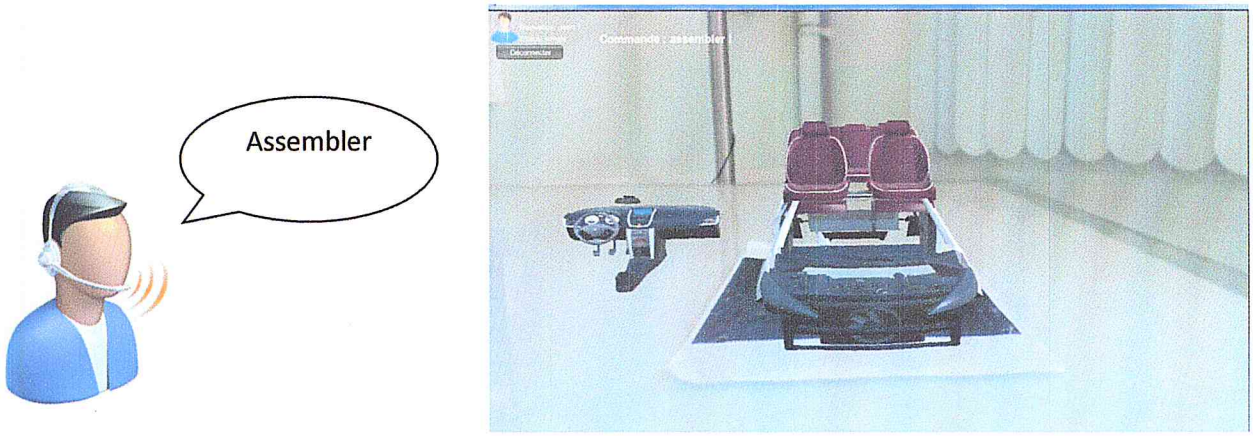


Figure 32 : *Composant 3D (siège) en cours d'assemblage.*

Figure 33 : *Fin d'assemblage de composant 3D (siège).*

6. Conclusion

Dans ce chapitre, nous avons d'abord montré l'architecture globale de notre système. Nous avons également présenté les environnements logiciels et matériels sur lesquels s'est basé notre travail, Ensuite, nous avons détaillé les composants essentiels de notre système. Enfin nous avons présenté nos tests et résultats.

Conclusion générale et perspectives

Conclusion générale

Dans ce projet de fin d'étude, notre mission a été de mettre en place un système capable de reconnaitre des commandes vocales simples, afin de contrôler des objets 3D dans un environnement de RA.

Au fil de ce mémoire, nous avons vu plusieurs aspects, nous avons commencé par une étude bibliographique sur la RA. Ensuite, nous avons présenté les différentes techniques d'interaction existantes dans la RA/RV. Une comparaison de ces techniques a été fait. Enfin, nous nous sommes recentrés sur l'aspect conceptuel et implémentation.

Ce projet a donné lieu à un système permettant d'abord d'augmenter une scène réelle par un ensemble d'objets 3D, dans notre cas, il s'agit des différents composants d'un véhicule. Ce système permet également à l'utilisateur d'interagir avec des objets virtuels en utilisant la voix. Ainsi, nous avons conçu et implémenté une solution complète capable de reconnaitre les commandes vocales que l'utilisateur prononce en utilisant un microphone. L'application que nous mettons à la disposition de cet utilisateur lui permet de sélectionner et manipuler les objets 3D superposés à la scène réelle de son environnement. Nous avons défini les commandes vocales de façon à permettre beaucoup de possibilités à l'utilisateur et pour avoir deux types de tâches d'interaction : la sélection et la manipulation (translation, rotation et zoom). En plus, nous avons utilisé l'interface clavier pour permettre à l'utilisateur de pouvoir contrôler les objets 3D par quelques touches clavier comme seconde modalité d'interaction.

En effet, l'interaction vocale est utilisée dans diverses applications parce qu'elle présente un certain nombre de points forts. Le principe consiste essentiellement à capter le signal de la parole et de le traduire en commandes appropriées. Les avantages perçus dans l'interaction vocale sont :

- Offrir une interaction naturelle à l'utilisateur.
- Libérer les mains de l'utilisateur.
- Réaliser une interaction à distance.

Lors de la réalisation de notre projet, nous avons été confrontés à plusieurs problèmes, parmi ceux, l'implémentation de la reconnaissance vocale sous Unity 3D.

Ce travail nous a, non seulement, donné l'opportunité de manipuler des technologies comme Unity3D, ARToolKit et de connaitre des nouveaux concepts comme la RA, mais aussi permis d'approfondir nos connaissances, en matière d'analyse et de conception des systèmes.

Perspectives

Dans le but d'enrichir notre système, certaines améliorations peuvent être ajoutées par la suite, elles sont définies comme suit :

- Utilisation d'une interaction multimodale, par exemple, la fusion de l'interaction vocale et l'interaction gestuelle à base de reconnaissance des gestes de la main.
- Implémentation de notre solution dans l'aspect collaboratif.
- Ajout des nouvelles fonctionnalités à notre système SCVRA.
- Intégration de lecteur vocal dans un environnement de réalité virtuelle.
- Cibler un autre domaine d'application que l'assemblage assisté par ordinateur.

Bibliographie

Bibliographie

[ART, 15] Getting started with ARToolKit for Unity, 2015, https://artoolkit.org/documentation/doku.php?id=6_Unity:unity_getting_started Consulté le 27 Mars 2016.

[ART2, 15] How does ARToolKit work, Mars 2015, https://www.hitl.washington.edu/artoolkit/documentation/userarwork.htm.

[ASL, 13] Alessandro L. Stamat to Ferreira, Leonardo Cunha de Miranda, Erica E. Cunha de Miranda, Sarah Gomes Sakamoto, A Survey of Interactive Systems based on Brain- Computer Interfaces, SBC Journal on 3D Interactive Systems, volume 4, number 1, 2013.

[AUG, 15] Key benefits of augmented reality for architecture projects, 2015, http://augmentedev.com/blog/key-benefits-augmented-reality-architecture-projects/, Consulté le 26/02/2016.

[AZU, 97] R.T. Azuma, A survey of augmented reality, Presence: Teleoperators and Virtual environments, vol. 6, no. 4, pp. 355–385, 1997.

[BEM, 04] Selim Benhimane and Ezio Malis. Real-time image-based tracking of planes using efficient second-order minimization, 2004.

[BER, 14] Berrached Chahrazed, Système de reconnaissance de gestes de main, Université Abou Bakr Belkaid–Tlemcen, 2014.

[BFO, 92] M. Bajura, H. Fuchs, and R. Ohbuchi. Merging virtual objects with the real world: seeing ultrasound imagery within the patient. In SIGGRAPH, pages 203–210, ACM, 1992.

[BLK, 81] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with and application to stereo vision. In Proceedings of the 7th International Joint Conference on Artificial Intelligence, Volume 2, IJCAI81, pages 674_679, San Francisco, CA, USA, 1981.

[BOU, 06] J. Bouchet, Ingénierie de l'interaction multimodale en entrée: approche à composants ICARE, Thèse de Doctorat, Grenoble 1, 2006.

[BOW, 99] D. A. Bowman, Interaction Techniques for Common Tasks in Immersive Virtual Environments: Design, Evaluation, and Application, Thèse de doctorat, Georgia Institute of Technology, 1999.

[BRM, 04] Bertrand David, René Chalon, Magali Beldame, OEil et IHM : suivi de regard et interaction à l'œil, 2004.

[BRU, 11] Aubin Bruno, Vers le développement d'un système interactif et collaboratif de réalité augmentée géo spatiale pour des applications en design urbain, Université Laval, 2011.

[BVB, 04] V. Buchmann, S. Violich, M Billinghurst, A Cockburn, FingARtips – Gesture Based Direct Manipulation in Augmented Reality, University of Canterbury, 2004.

[CHC, 11] M Chang, W. Y. Hwang, M. P. Chen, W. Mueller, Edutainment Technologies Educational Games and Virtual Reality/Augmented Reality Applications, 6th International Conference on E-learning and Games, Edutainment, Taipei, Taiwan, Proceedings, vol. 6872, Springer Science & Business Media, 2011.

[CON, 08] Simon Conseil, Suivi tridimensionnel de la main et reconnaissance de gestes pour les Interfaces Homme Machine, Thèse de doctorat, université Paul Cézanne Aix- Marseille 3, 2008.

[COT, 97] M. K. D. Coomans, H. J. Timmermans, Towards a taxonomy of virtual reality user interfaces, In Proceedings of IEEE Conference on Information Visualization, pp. 279–284, 1997.

[DEH, 08] Christophe Dehais, Contribution pour les applications de réalité augmentée. Suivi visuel et recalage 2D. Suivi d'objets 3D représentés par des modèles par points, Institut de Recherche en Informatique de Toulouse, May 2008.

[DEL, 13] Florent Delomier, Jeux pédagogiques collaboratifs situés : conception et mise en œuvre dirigées par les modèles, Ecole centrale de Lyon, Décembre 2013.

[DEV, 14] Créer un jeu avec Unity 3D, 2014, http://jeux.developpez.com/videos/tutoriels/unity/02-interface/ , Consulté le 29 Mai 2016.

[DJE, 13] Fahima Djelil, Étude et évaluation de l'apport des NUIs pour les applications de réalité virtuelle et augmentée, Université d'Évry Val d'Essonne, Thèse de master, Juin 2013.

[DOM, 09] Jean-Yves Didier, Samir Otmane, and Malik Mallem.Arcs : Une architecture logicielle reconfigurable pour la conception des applications de réalité augmentée. Technique et Science Informatiques (TSI), Réalité Virtuelle-Réalité Augmentée, Juin septembre 2009.

[FMS, 93] Steve Feiner, Blair Macintyre, and D. Seligman. Knowledge-based augmented reality. Communications of the ACM, 36(7) : 52–62, July 1993.

[GAU, 05] G. Gauffre, Techniques d'interaction augmentée: vers un lien entre modélisation et conception logicielle, Mémoire de Master , Université Paul Sabatier, Toulouse III, 2005.

[GLW, 96] W.E.L. Grimson, T. Lozano-Perez, W.M. Wells III, G.J. Ettinger, S.J. White, and R. Kikinis. An automatic registration method for frameless stereotaxy, image guided surgery, and enhanced reality visualization. *Transactions on Medical Imaging*, 1996.

[GON, 14] Frédéric Goncalves, Conception d'un environnement virtuel avec adaptation de l'immersion pour la simulation de conduite en fauteuil roulant, Laboratoire d'ingénierie des systèmes de Versailles, 2014.

[GUE, 13] Guemougui Abdessattar, Réalité augmentée pour les jeux vidéo et les serious games, Université Mohamed Khider–Biskra, 2013.

[HAG, 12] Claire HAGRY , Les enjeux de la réalité augmentée Genève, le 21 décembre 2012, Haute École de Gestion de Genève (HEG-GE), Filière Informatique de Gestion

[HAY, 11] M. Haydar, Interaction en réalité mixte appliquée à l'archéologie sous-marine, Doctoral dissertation, Evry-Val d'Essonne, 2011.

[HIL, 13] Nicolas Hilaire, Apprenez à développez en c#, 2013, https://openclassrooms.com/courses/apprenez-a-developper-en-c, Consulté le 23 Mai 2016.

[HME, 07] H. Mercier, Modélisation et suivi des déformations faciales : Applications à ladescription des expressions du visage dans le contexte de la langue des signes, Doctoral dissertation, Université Paul Sabatier-Toulouse III, 2007.

[HNL, 96] William A. Ho, Khoi Nguyen, and Torsten Lyon.Computer vision-based registration techniques for augmented reality. In Intelligent Robots and Computer Vision XV, pages 538_548, 1996.

[IHU, 15] IHU de Strasbourg, Une chirurgie robotisée guidée par la Réalité Augmentée, 30 Septembre 2014, http://www.ircad.fr/fr/ihu-de-strasbourg-quatre-premieres-en-moins-dun-an-dexistence/, Consulté le 14 Décembre 2015.

[INI, 11] FORD C-MAX launch outdoor augmented reality compaign, 17 Février 2011, http://www.inition.co.uk/ford-c-max-launch-outdoor-augmented-reality-campaign/, Consulté le 13 Janvier 2016.

[IRC, 11] Dictionnaire IRCOM, 2011, http://ircom.huma-num.fr/site/glossaire.php, Consulté le 25 Janvier 2016.

[KAB, 99] H. Kato, M. Billinghurst, Marker tracking and hmd calibration for a video-based augmented reality conferencing system, In Proceedings of 2nd IEEE and ACM International Workshop on Augmented Reality, (IWAR'99), pp, 85–94, 1999.

[LAR, 11] F. Larrue, Influence des interfaces dans le transfert du virtuel au réel, Thèse de Doctorat, Université Bordeaux 2, 2011.

[LEB, 11] D. Leblanc, Réalité augmentée en robotique, 25 Mai 2011, http://shyrobotics.com/realite-augmentee-en-robotique_20110525.html, Consulté le 14 Décembre 2015.

[LUS, 07] Blaise Lusikila, Conception et implémentation d'une base de données pour la gestion d'un organisme et administration réseau à distance sur base des outils libres, Université Cardinal Malula, 2007.

[LVT, 03] Vincent Lepetit, Luca Vacchetti, Daniel Thalmann, and Pascal Fua.Fully automated and stable registration for augmented reality applications. In Proceedings of the 2Nd IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR 03, Washington, DC, USA, 2003. IEEE Computer Society.

[MAC, 96] W. E. Mackay, Réalité Augmentée : le meilleur des deux mondes, La Recherche, no. 285, pp. 32–37, 1996.

[MAR, 08] Malik Mallem et David Roussel. Réalité augmentée : Principes, technologies et applications. Ed. Techniques Ingénieur, Référence TE5920, 2008.

[MCC, 14] James McCaffrey, Reconnaissance de la parole grâce aux applications .NET Desktop, 2014 https://msdn.microsoft.com/fr-fr/magazine/dn857362.aspx, Consulté le 22 Mai 2016.

[MCW, 00] D.R. McGee, P.R. Cohen, et L.Wu (2000). Something from nothing: augmenting a paper-based work practice via multimodal interaction. Dans Proceedings of DARE 2000 on Designing augmented reality environments, pages 71–80. ACM New York, NY, USA, 2000.

[MED, 03] M. Schneider, B. Schwald, H. Seibert et T. Weller, Medarpa, a medical augmented reality system for minimal-invasive interventions, Studies in health technology and informatics, n° 94, 2003: pp. 312-314.

[MIK, 94] P. Milgram and F. Kishino. A taxonomy of mixed reality visual displays. In IEICE Transactions on Information and Systems (Special Issue on Networked Reality, number E77-D(12), December 1994.

[MIN, 02] W. Minker, F. Néel, Développement des technologies vocales, Presses Universitaires de France (PUF), vol. 65, no. 3, pp. 261–287, 2002.

[MLD, 06] E. Mouragnon, Maxime Lhuillier, Michel Dhome, Fabien Dekeyser, and Patrick Sayd. Monocular vision based slam for mobile robots. In ICPR (3), pages 1027_1031,IEEE Computer Society, 2006.

[NIS, 03] David Nistér. Preemptive ransac for live structure and motion estimation. In Proceedings of the Ninth IEEE International Conference on Computer Vision – Volume 2, ICCV '03, pages 199_, Washington, DC, USA, 2003. IEEE Computer Society.

[ORT, 07] Jean-José Orteu, Calibrage géométrique d'une caméra ou d'un capteur de vision stéréoscopique, 2007, http://www.optique-ingenieur.org/fr/cours/OPI_fr_M04_C01/co/OPI_fr_M04_C01_web_1.html, Consulté le 03 Janvier 2016.

[PFE, 08] Pfeiffer, T Towards gaze interaction in immersive virtual reality: evaluation of a monocular eye tracking set-up. In Virtuelle und Erweiterte Realität -Fünfter Workshop der GI-Fachgruppe, (2008).

[PFM, 06] Phillipe Fuchs and Guillaume Moreau. Le traité de la réalité virtuelle : Volume 1, l'homme et l'environnement virtuel. Presses de l'Ecole des mines, 2006.

[PHP, 01] PHP, 2001, http://php.net/manual/fr/introwhatis.php, Consulté le 24 Mai 2016.

[PIP, 06] Dan pilone, Neil Pitman, UML2.0 en concentré, Edition O'reilly, page 222, 2005.

[PRM, 05] M. Pressigout and E. Marchand. Real time planar structure tracking: A contour and texture approach. Technical Report 1698, IRISA, March 2005.

[PRM, 06] M. Pressigout and E. Marchand. Real-time 3d model-based tracking : Combining edge and texture information. In IEEE Int. Conf. on Robotics and Automation, ICRA'06, pages 2726_2731, Orlando, Florida, May 2006.

[ROB, 92] W. Robinett, Synthetic Experience: A proposed Taxonomy, Presence: Teleoperators and Virtual Environments, vol. 1, no. 2, pp. 229–247, 1992.

[ROQ, 06] P.Roques, UML 2 en partique, Edition Eyrolles, 2006.

[ROV, 07] Pascal Roques, Franck Vallée, UML 2 en action de l'analyse des besoins à la conception, 4e édition, février 2007.

[SLH, 96] Andrei State, Mark A. Livingston, Gentaro Hirota, William F. Garrett, Mary C. Whitton, HenryFuchs, and Etta D. Pisano (MD). Technologies for augmented-reality systems : realizing ultrasound-guided needle biopsies, ACM, In SIGGRAPH, pages 439–446, New Orleans, USA, August 1996.

[SNT, 98] Y. Sato, M. Nakamoto, Y. Tamaki, T. Sasama, I. Sakita, Y. Nakajima, M. Monden, and S. Tamura. Image guidance of breast cancer surgery using 3-d ultrasound images and augmented reality visualization.Medical Imaging, IEEE Transactions on, 17(5), pages 681_693, 1998.

[STJ, 15] Stéphan Julienne, Avec le eye-tracking, un coup d'œil suffit pour jouer en toute liberté, 2012, http://www.futurimmediat.fr/eye-tracking-coup-d-oeil-pour-jouer-toute-liberte/, Consulté le 04 février 2016

[SUT, 65] Yvan E. Sutherland. The ultimate display. In IFIPS Congress, volume 2, pages 506–508, NewYork, USA, May 1965.

[SUT, 68] Yvan E. Sutherland. A head-mounted three-dimensional display. In AFIPSConference, volume33, pages 757–764, 1968.

[TAR, 05] Jérome Tarniewicz, Etude d'une méthode de sondage de la vapeur d'eau dans la troposphére appliquée à la correction de mesures GPS pour l'altimétrie de haute précision, Ecole doctorale Sciences de l Environnement d'Ile de France, mars 2005.

[UNI, 15] Unity3D, 2015. http://docs.unity3d.com/Manual/CreatingAndUsingScripts.html, consulté le 29 Mai 2016.

[VIR, 16] Qu'est-ce que la RA ?, 2016, http://www.realite-virtuelle.com/definition-realite-augmentee, Consulté le 07 Février 2016.

[VLF, 04] Luca Vacchetti, Vincent Lepetit, and Pascal Fua.Combining edge and texture information for real-time accurate 3d camera tracking.In In IntSymp on Mixed and Augmented Reality, 2004.

[WEL, 93] Pierre Wellner. Interacting with paper on the Digital Desk. Commun.ACM, vol. 36, no. 7, pages 87–96, ACM, New York, NY, USA, jul 1993.

[WEB, 13] Les Google Glass, 2013, http://www.webrankinfo.com/google/glass.htm, Consulté le 23 Janvier 2016.

[WFM, 96] A. Webster, Steve Feiner, Blair MacIntyre, W. Massie, and T. Krueger. Augmented reality in architectural construction, inspection and renovation. In Congress on Computing in Civil Engineering, pages 913–919, Anaheim, USA, June 1996. ASCE.

[ZEF, 12] Marc Zaffani, Nouvelle avancée dans le contrôle de robots par la pensée, Futura sciences , 2012, http://www.futura-sciences.com/ , Consulté le 02 Février 2016.

[ZEN, 10] Iman Mayssa Zendjebil. Localisation 3D basée sur une approche de suppléance multi-capteurs pour la réalité augmentée mobile en milieu extérieur. Evry-Val d'Essonne, Octobre 2010.

Glossaire

2TUP	Two Track Unifiend Process
3D	3Dimensions
API	Application Programming Interface
ARTo	Augmented Reality Toolkit
BCI	Brain Computer Interface
CAO	Conception Assistée par Ordinateur
CCC	Concentric Contrasting Circle
CDTA	Centre de Développement des Technologies Avancées
EV	Entités Virtuelles
GPS	Global Positioning System
IND	Interface Neuronale Directe
IR	Images Réelles
IRM	Imagerie par Résonnance Magnétique
IRVA	Interaction Homme Machine Réalité Virtuelle et Augmentée
JRL	Joint Robotics Laboratory
KARMA	Knowledge-based Augmented Reality for Maintenance Assistance
MMS	Multimedia Messaging Service
MySQL	My Structured Query Language
PHP	Hypertext Preprocessor
RA	Réalité Augmentée
RA	Réalité Mixte
RFID	Radio Frequency Identification
RV	Réalité Virtuelle
SAPI	Speech Application Programming Interface
SCVRA	Système de Commande Vocale en Réalité augmentée
SFM	Structure From Motion

SGBD	Système de Gestion de Bases de Données
SIG	Système d'Information Géographique
SLAM	Simultaneous Localization And Mapping
SMS	Short Message Service
SQL	Structured Query Language
UML	Unified Modeling Language
VA	Virtualité Augmentée

Annexes

Annexe A

Calibrage de caméra

A.1. Calibrage de caméra

Le calibrage géométrique d'une caméra consiste à déterminer la relation mathématique existante entre les coordonnées des points 3D de la scène observée et les coordonnées 2D de leur projection dans l'image (points-image). Calibrer une caméra, c'est choisir un modèle de caméra a priori et déterminer ensuite les paramètres de ce modèle [ORT, 07]. On note que les termes : étalonnage et calibration (anglicisme) sont aussi couramment utilisés.

A.2. Modélisation d'une caméra

Il existe plusieurs modèles de caméras, le plus simple est le modèle du sténopé, dit aussi de la chambre noire. Il s'agit d'un modèle idéal couramment utilisé en vision par ordinateur pour exprimer le principe de fonctionnement de la caméra [GUE, 13].

A.2.1 Modèle de sténopé

Le modèle sténopé ("pinhole" en anglais) modélise une caméra par une projection perspective. Ce modèle suppose que le système optique de la caméra, c'est-à-dire sa lentille respecte les conditions de Gauss [GUE, 13]. Il transforme un point 3D de l'espace M en un point-image m. Principalement le processus peut se décomposer en trois transformations élémentaires successives (comme illustré sur la figure A.1) [ORT, 07] :

1. La transformation entre le repère du monde et celui de la caméra.
2. La transformation entre le repère caméra et le repère capteur (plan rétinien).
3. La transformation entre le repère capteur et le repère image.

Il convient de préciser les repères orthonormés employés pour la calibration de la caméra :

- (Rw,$\overrightarrow{Xw}$,$\overrightarrow{Yw}$,$\overrightarrow{Zw}$) : repère associé à l'espace de travail (repère du monde). S est l'origine de ce repère.
- (Rc, $\overrightarrow{Xc}$, $\overrightarrow{Yc}$, $\overrightarrow{Zc}$) : repère associé à la caméra. C est le centre optique de la caméra.
- (Rr, $\vec{u}$, $\vec{v}$) : repère associé à l'image visualisée.

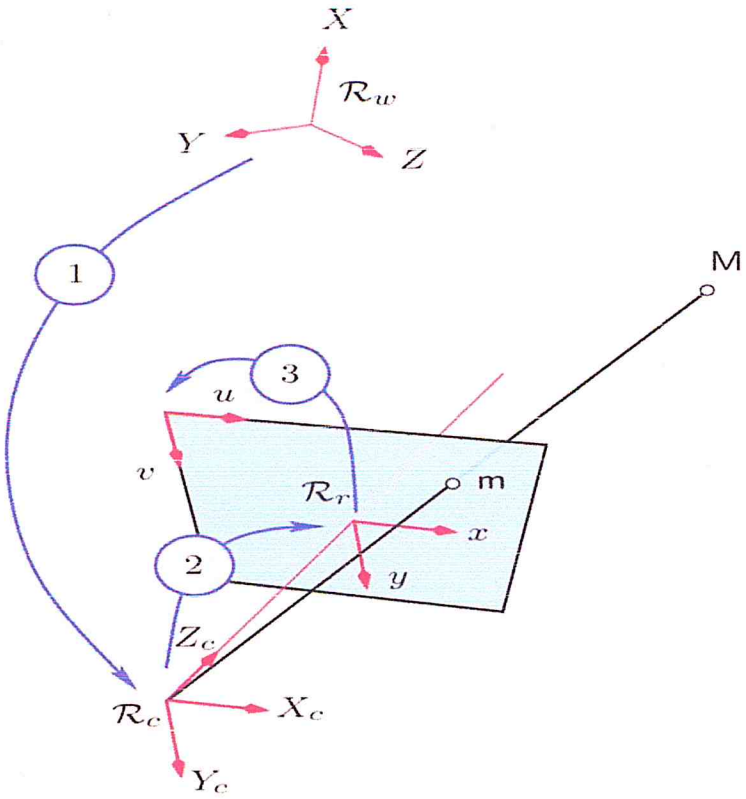


Figure A.1 : *Les transformations élémentaires du modèle sténopé [ORT, 07].*

Les paramètres employés dans le modèle sténopé sont usuellement divisés en deux catégories, les paramètres intrinsèques qui sont internes à la caméra, et les paramètres extrinsèques qui peuvent varier suivant la position de la caméra dans l'espace de travail [GUE, 13].

A.2.1.1. Paramètres extrinsèques

Soit le point M de coordonnées (X, Y, Z) ^{T} exprimées dans le repère monde. Ses coordonnées dans le repère caméra (Xc, Yc, Zc) ^{T} sont obtenues à partir de la relation suivante :

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = R \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} + t = [R|t] \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \qquad (A.1)$$

Où (X, Y, Z, 1) ^{T} sont les coordonnées homogènes du point P dans le repère monde. R et t représentent le déplacement rigide entre les deux repères, R étant la matrice de rotation et t le vecteur de translation. Ces paramètres définissent la position et l'orientation de la caméra par rapport au repère monde [ZEN, 10].

A.2.1.2 Paramètres intrinsèques

La projection du point M de coordonnées (Xc, Yc, Zc) ^{T} définies dans le repère caméra, est le point m de coordonnées (Xc, Yc, Zc) ^{T} définies dans le repère caméra, telle que :

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = f \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (A.2)$$

Dans le repère image 2D, le point m est exprimé en coordonnées pixel (u, v) ^{T} obtenues grâce à la relation suivante :

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} k_u & -k_v \cos\theta & u_0 \\ 0 & -k_v \sin\theta & v_0 \end{pmatrix} \begin{pmatrix} x_c \\ y_c \\ 1 \end{pmatrix} \quad (A.3)$$

Où k_u et k_v représentent les distances focales exprimées en pixels suivant l'axe $\vec{u}$ et l'axe $\vec{v}$ respectivement. $(u_0, v_0)^T$ sont les coordonnées pixel du point centre c. θ est l'angle entre les deux axes (u et v) dit aussi angle de distorsion. Les paramètres k_u, k_v, f, u_0, v_0, θ sont les paramètres intrinsèques de la caméra [ZEN, 10].

Annexe B

ARToolKit

La présente annexe comprendre deux parties, une première là où nous présenterons la librairie logicielle ARToolKit et une seconde consacrée aux étapes que nous avons utilisé pour la création d'une scène augmentée.

Partie 1 :

B.1. Présentation d'ARToolKit

ARToolkit (Augmented Reality ToolKit) est une bibliothèque logicielle développée en C et C++, open-source et multiplateformes, utilisée pour concevoir des applications de RA. Elle a été développée par le docteur Hizokazu Kato à l'Université d'Osaka en 1999.

ARToolKit utilise les techniques de vision par ordinateur pour calculer, en temps réel, la position et l'orientation de caméra réelle par rapport à des marqueurs. Cette librairie logicielle supporte plusieurs types de marqueurs. Le suivi rapide et précis fourni par ARToolKit a permis le développement rapide de milliers des applications intéressantes en RA.

B.2. Principe de fonctionnement d'ARToolKit

L'algorithme de détection d'ARToolkit effectue une suite d'opérations sur chaque image capturée du flux vidéo afin, d'une part de repérer la présence d'un marqueur, puis de l'identifier parmi les différents marqueurs chargés dans l'application (dans le cas d'une application multi-marqueurs). Les principales étapes d'exécution d'ARToolKit sont :

- La caméra capture la vidéo et l'envoie vers l'ordinateur.
- ARToolKit binarise l'image, puis recense toutes les formes carrées (cadres noirs) présentes dans chaque image de la vidéo.
- Pour chaque forme carrée détectée, le programme utilise des formules mathématiques pour déterminer la position et l'orientation de la caméra par rapport au carré noir.
- Le motif présent à l'intérieur de chaque cadre est comparé avec les motifs chargés dans le programme afin de connaitre l'augmentation qui lui est associée.
- L'augmentation, qui prend généralement la forme d'un objet 3D, est alors générée.

- Cette augmentation est ensuite superposée à l'image capturée, positionnée, orientée et mise à l'échelle suivant les données fournies par le marqueur.

La figure ci-dessous résume ces étapes :

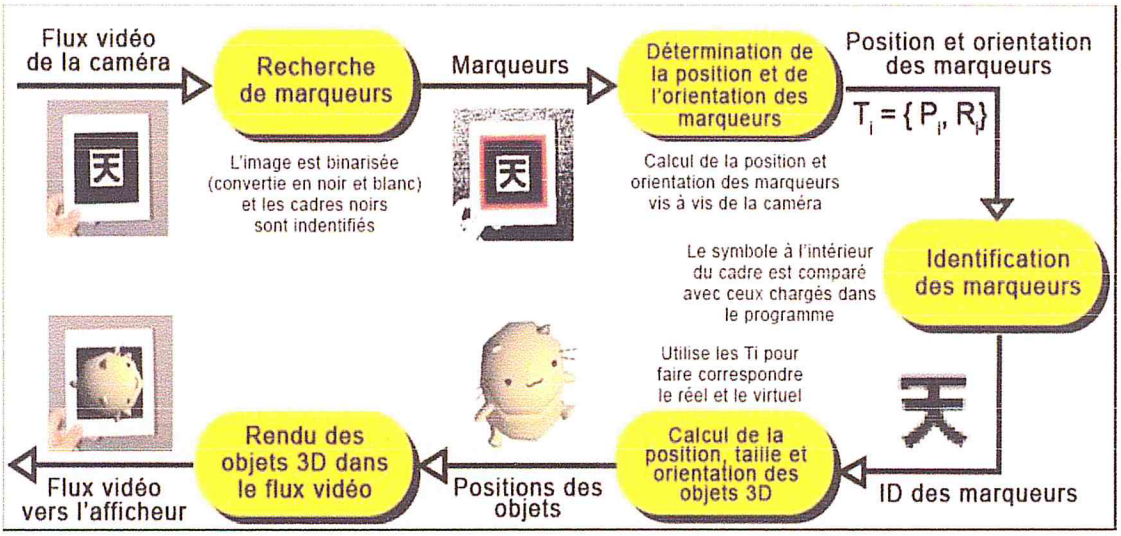


Figure B.1 : *Principe de fonctionnement d'ARToolKit.*

B.3. Limitation d'ARToolKit

Alors que le suivi basé sur la vision est excitant pour permettre tant d'applications, il existe des limitations qui affectent ARToolKit et d'autres systèmes basés sur la vision :

Les objets virtuels apparaissent uniquement lorsque le marqueur étant suivi, est en vue de la caméra. Cela peut limiter la taille ou le mouvement des objets virtuels et signifie également que si les utilisateurs couvrent une partie de motif avec leurs mains ou avec d'autres objets, l'objet virtuel disparaîtra.

Les conditions d'éclairage influencent aussi sur les résultats du suivi. Les lumières aériennes peuvent créer des réflexions et des taches d'éblouissement sur un marqueur de papier, ce qui rend la tâche de trouver ce dernier plus difficile [ART2].

Partie 2 :

B.4. ARToolKit pour Unity 3D

ARToolKit pour Unity 3D est distribué comme un package qui contient les plugins, les scripts et les ressources nécessaires pour intégrer ARToolKit avec une application Unity 3D. Pour implémenter ARToolKit pour Unity 3D sous Unity 3D, nous avons besoin de créer des objets vides qui vont contenir les objets de configuration de réalité augmentée. Parmi ces objets de configurations nous citons : ARController, ARMarkers, AROrigin, ARTrackedObjects et ARCamera.

B.5. Configuration de la scène de réalité augmentée

Voici, les différentes étapes qu'on a suivi pour créer une scène de réalité augmentée sous Unity 3D :

B.5.1. ARController

D'abord, il faut créer un objet vide (Empty GameObject). Une fois créé, nous le renommons "ARController" (le nom de cet objet n'a aucune incidence sur la configuration de la scène, donc nous pouvons renommer l'objet comme nous voulons), Ensuite nous glissons le script ARController dans cet objet.

Figure B.2 : *glissement de script ARController dans l'objet vide crée.*

Nous devons maintenant être en mesure d'exécuter la scène et de voir la vidéo en direct.

B.5.2. ARMarkers

Le suivi requis des marqueurs, donc il faut glisser autant de script ARMarker que nous souhaitons suivre dans le 1er objet vide créé "ARController". Dans notre cas, nous utilisons un seul marqueur par conséquent nous glissons un seul script ARMarker dans l'objet vide "ARController".

Maintenant, nous configurons ARMarker, en étant sûr de définir son champ "Tag" (dans notre cas "Marker"). En effet, "Marker" sera le nom utilisé par la scène pour trouver le marqueur et obtenir ses données.

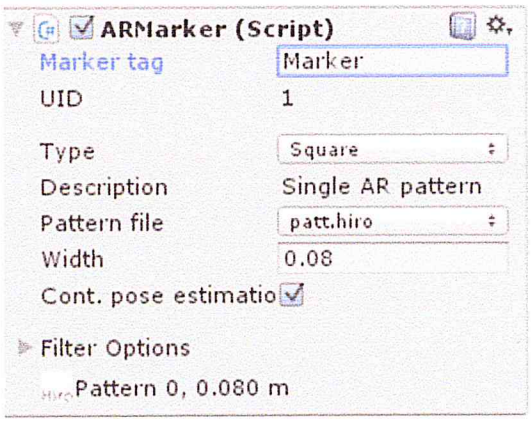


Figure B.3 : *Définition de champ Tag de ARMarker.*

B.5.3. AROrigin

A ce stade, nous choisissons le point {0, 0, 0} dans notre graphe de scène pour qu'il soit la racine de notre scène et nous créons un autre objet vide à cette racine. Une fois créé, nous renommons cet objet "Scene root". Aprés cela nous glissons un script AROrigin sur cet objet.

Il est utile de mettre l'objet "Scene root" et tous ses enfants dans sa propre couche, pour cela, nous avons créé une nouvelle couche (Layer) "AR Background" pour l'objet "Scene root".

Nous notons que ARCamera et chaque ARTrackedObject (nous parlons de ces objets de configurations dans ce qui suit) devraient être des enfants de cet objet. Cela permet de quelques avantages:

- Tous les objets interagissent dans le même espace de coordonnées.
- Recherche d'objets par type devient beaucoup moins cher une fois que nous savons qui est le parent.

B.5.4. ARTrackedObjects

A ce moment, nous ajoutons un objet enfant sous la racine de la scène que nous avons créé à l'étape précédente (l'objet "Scene root") et nous renommons "Marker Scène". Ensuite, il faut attachez le script ARTrackedObject à cet objet et configurer son propriété "Tag" au même nom que nous avons utilisé sur son ARMarker correspondant. Cela va associer ARTrackedObject avec ARMarker.

B.5.5. ARCamera

La dernière chose que nous devons ajouter avant que le contenu peut être affiché, est un objet Main Camera d'Unity 3D . Cet objet doit être un fils de l'objet "Scene root", il faut définir aussi le masque d'abattage (culling mask) de caméra à la couche "AR Background". Une fois le masque d'abattage est défini, nous attachons un script ARCamera sur l'objet Main Camera.

B.5.6. Test et ajout de contenu

À ce stade, nous exécutons à nouveau la scène, les messages de console apparaissent et notifient que le marqueur a été trouvé ou perdu bien qu'aucun contenu apparaît sur le marqueur.

Par défaut, les marqueurs apparaissent dans la scène virtuelle verticalement (comme un panneau d'affichage) à l'origine, pour que le marqueur repose à plat sur le sol de la scène virtuelle (figure B.4), nous devons changer dans l'inspecteur de l'objet "Scene root" sa "Rotation: X" à la valeur 90. Cela va faire pivoter tous les objets fils de l'objet "Scene root" de 90 degrés autour de l'axe X.

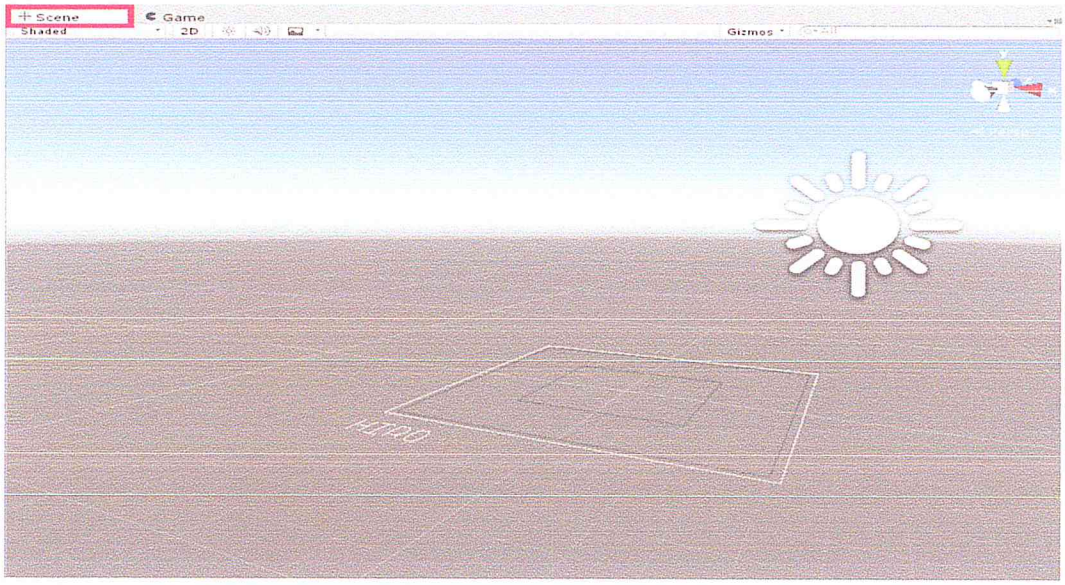


Figure B.4 : *Marqueur apparait horizontalement dans la scène virtuelle.*

Maintenant, nous glissons l'objet 3D "voiture" sur l'objet "Marker scene", cela signifie que l'objet 3D "Voiture" devient un fils de l'objet "Marker scene". Ensuite nous réglons l'échelle et la position de l'objet 3D "Voiture" pour que cet objet soit siégé sur le marqueur correctement. Enfin nous changeons la couche (Layer) de l'objet 3D "Voiture" à "AR Background" (nous appliquons le changement à tous les enfants de l'objet 3D "Voiture" lorsque nous somme invités).

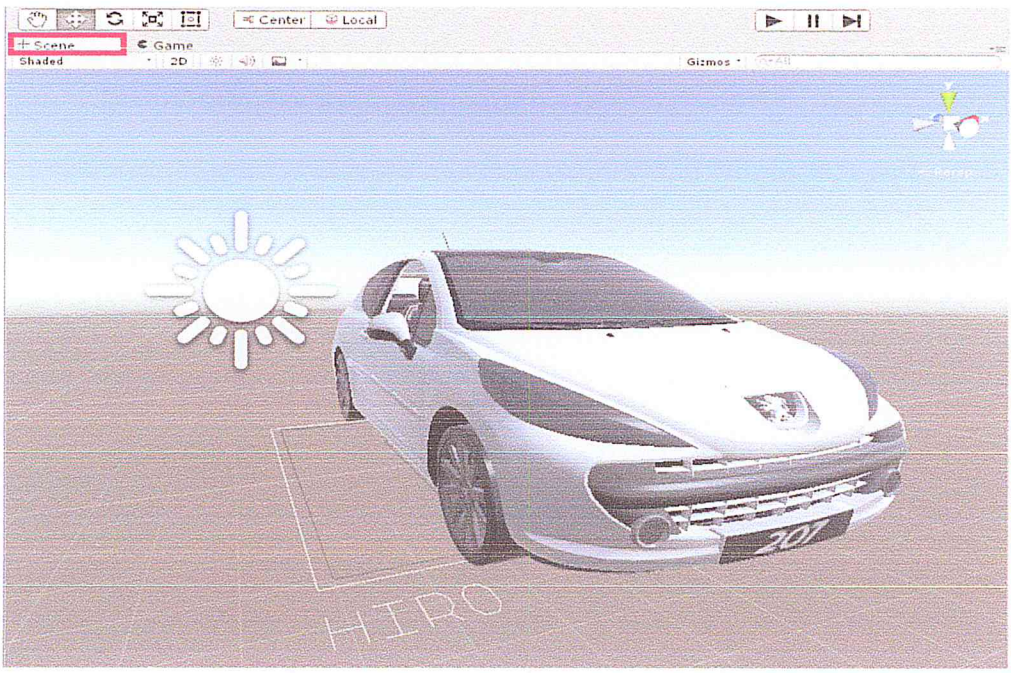


Figure B.5 : *Ajout de l'objet 3D voiture.*

B.5.7. Choisir entre plusieurs webcams

En utilisant le module vidéo par défaut (WinMc), nous pouvons commander que ARToolKit pour Unity 3D utilise une deuxième webcam ou source d'entrée vidéo, en ajoutant

devNum = 2 dans la boîte de dialogue de configuration vidéo.

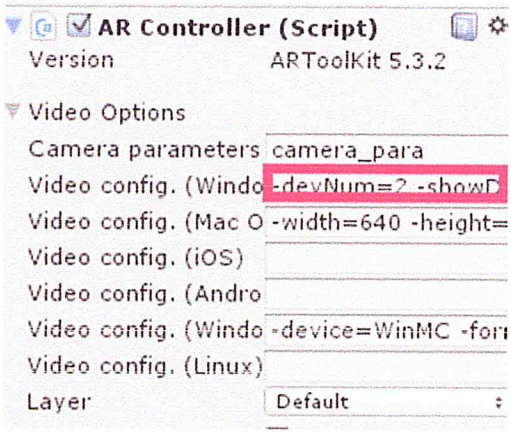


Figure B.6 : *choisir entre plusieurs webcams dans la boite de configuration vidéo.*

Annexe C

Processus de développement logiciel et langage de modélisation

C.1. Processus de développement logiciel

Un processus de développement définit une séquence d'étapes, en partie ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant. Son objectif est de produire des logiciels de qualité qui répondent aux besoins de leurs utilisateurs dans des temps et des couts prévisibles [ROV, 07] (figure C.1).

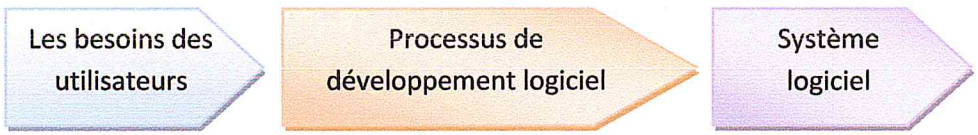


Figure C.1 : *Processus de développement logiciel.*

Nous pouvons citer à ce propos, les processus de développement logiciel suivant : 2TUP, RUP, AUP, XP, etc.

C.2. Le processus 2TUP

2TUP signifie « 2 Track Unified Process ». C'est un processus UP qui répond aux caractéristiques que nous venons de citer. Le processus 2TUP apporte une réponse aux contraintes de changement continuel imposées aux systèmes d'information de l'entreprise. En ce sens, il renforce le contrôle sur les capacités d'évolution et de correction de tels systèmes. « 2 Track » signifie littéralement que le processus suit deux chemins : des chemins « fonctionnels » et « d'architecture technique », qui correspondent aux deux axes de changement imposés au système informatique [ROV, 07].

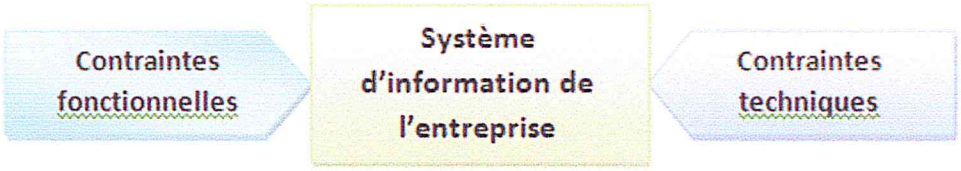


Figure C.2 : *Le système d'information soumis à deux natures de contraintes [ROV, 07].*

À l'issue des évolutions du modèle fonctionnel et de l'architecture technique, la réalisation du système consiste à fusionner les résultats des deux branches. Cette fusion conduit à l'obtention d'un processus de développement en forme de Y, comme illustré par la figure C.3.

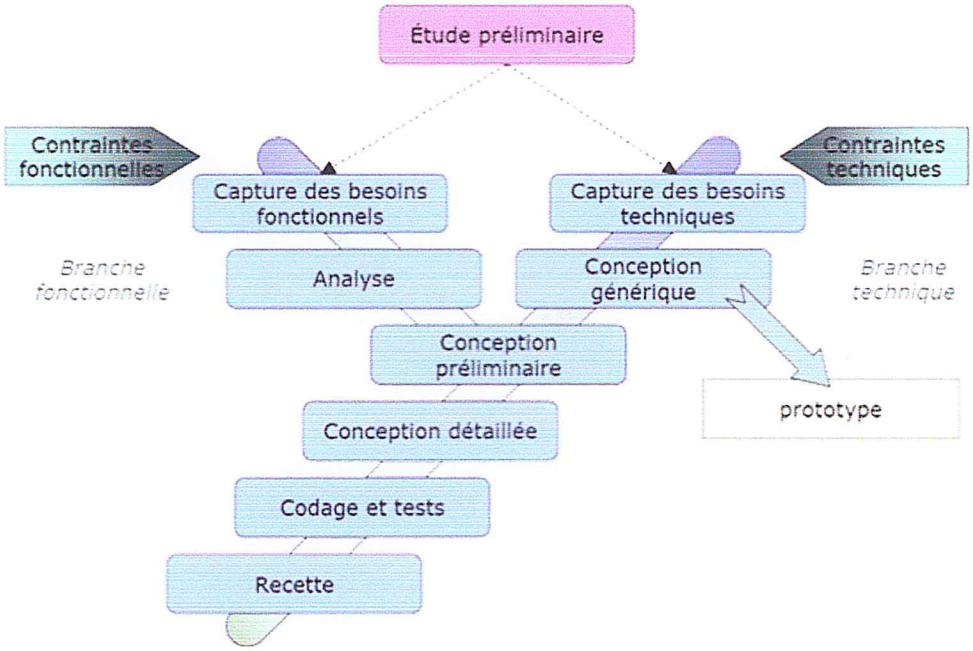


Figure C.3 : *Le processus de développement en Y [ROV, 07].*

Le déroulement du processus 2TUP est résumé dans les étapes suivantes :

- **La branche gauche (fonctionnelle) comporte** :
  - ***la capture des besoins fonctionnels*** : qui produit un modèle des besoins focalisé sur le métier des utilisateurs. Elle qualifie au plus tôt le risque de produire un système inadapté aux utilisateurs. De son côté, la maîtrise d'œuvre consolide les spécifications et en vérifie la cohérence et l'exhaustivité de l'analyse.

- ***L'analyse*** : qui consiste à étudier précisément la spécification fonctionnelle de manière à obtenir une idée de ce que va réaliser le système en termes de métier. Les résultats de l'analyse ne dépendent d'aucune technologie particulière.

➢ **La branche droite (architecture technique) comporte :**

- ***la capture des besoins techniques*** : qui recense toutes les contraintes et les choix dimensionnant la conception du système. Les outils et les matériels sélectionnés ainsi que la prise en compte de contraintes d'intégration avec l'existant conditionnent généralement des pré-requis d'architecture technique.

- ***la conception générique*** : qui définit ensuite les composants nécessaires à la construction de l'architecture technique. Cette conception est la moins dépendante possible des aspects fonctionnels. Elle a pour objectif d'uniformiser et de réutiliser les mêmes mécanismes pour tout un système.

L'architecture technique construit le squelette du système informatique et écarte la plupart des risques de niveau technique. L'importance de sa réussite est telle qu'il est conseillé de réaliser un prototype pour assurer sa validité.

➢ **La branche du milieu comporte :**

- ***la conception préliminaire*** **:** qui représente une étape délicate, car elle intègre le modèle d'analyse dans l'architecture technique de manière à tracer la cartographie des composants du système à développer.

- ***la conception détaillée*** **:** qui étudie ensuite comment réaliser chaque composant.

- ***l'étape de codage :*** qui produit ces composants et teste au fur et à mesure les unités de code réalisées.

- ***l'étape de recette*** **:** qui consiste enfin à valider les fonctions du système développé.

C.3. Le langage de modélisation UML

UML est l'acronyme de << Unified Modeling Langage >>, il se définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier, concevoir des solutions et communiquer des points de vue [PIP, 06].

UML unifie à la fois les notations et les concepts orientés objet, il ne s'agit pas d'une simple notation mais les concepts transmis par un diagramme ont une sémantique précise et sont porteurs de sens au même titre que les mots d'un langage, c'est pour ça qu'UML est présenté parfois comme une méthode alors qu'il ne l'est absolument pas.

UML unifie également les notations aux différentes activités d'un processus de développement et offre, par ce biais, le moyen d'établir le suivi des décisions prises, depuis la définition des besoins jusqu'au codage [PRO, 06].

Voici une présentation rapide des différents diagrammes UML qui vont être utilisés tout au long de notre projet :

- **Le diagramme des cas d'utilisation :** représente la structure des fonctionnalités nécessaires aux utilisateurs du système. Il est normalement utilisé lors des étapes de capture des besoins fonctionnels et techniques.

- **Le diagramme de classes :** surement l'un des diagrammes les plus importants dans un développement orienté objet. Sur la branche fonctionnelle, ce diagramme est prévu pour développer la structure des entités manipulées par les utilisateurs. En conception, le diagramme de classe représente la structure d'un code orienté objet.

- **Le diagramme de séquence :** représente les échanges de messages entre objets, dans le cadre d'un fonctionnement particulier du système.

C.4.1. Description du contexte

A. Identification des messages acteurs / système :

Nous détaillons dans le tableau suivant les différents messages échangés entre les acteurs et le système :

Message	Description des messages	Expéditeur	Destinataire
Msg1	Demande d'authentification	- Administrateur - Expert - Utilisateur simple	- SCVRA
Msg2	Accès à la session personnelle	- SCVRA	- Administrateur - Expert - Utilisateur simple
Msg3	Message d'erreur : Login ou mot de passe incorrect	- SCVRA	- Administrateur - Expert - Utilisateur simple
Msg4	Etablissement des préférences	- Administrateur - Expert - Utilisateur simple	- SCVRA
Msg5	Session modifiée selon les préférences	- SCVRA	- Administrateur - Expert - Utilisateur simple
Msg6	Demander l'affichage d'une famille de composants 3D	- Expert - Utilisateur simple	- SCVRA
Msg7	Afficher les composants 3D de cette famille	- SCVRA	- Expert - Utilisateur simple
Msg8	Sélectionner ou désélectionner un composant 3D	- Expert - Utilisateur simple	- SCVRA
Msg9	composant sélectionné ou désélectionné	- SCVRA	- Expert - Utilisateur simple
Msg10	Demande d'assemblage d'un composant 3D	- Expert - Utilisateur simple	- SCVRA
Msg11	Composant assemblé	- SCVRA	- Expert - Utilisateur simple
Msg12	Demande de manipulation d'un composant 3D	- Expert - Utilisateur simple	- SCVRA
Msg13	Composant manipulé	- SCVRA	- Expert - Utilisateur simple
Msg14	Demande de mise à jour d'un composant 3D	- Expert	- SCVRA
Msg15	Interface de mise à jour de composant 3D	- SCVRA	- Expert
Msg16	Mise à jour de composant 3D	- Expert	- SCVRA
Msg17	Demande de mise à jour d'un compte utilisateur	- Administrateur	- SCVRA

Msg18	Interface de mise à jour de compte utilisateur	- SCVRA	- Administrateur
Msg19	Mise à jour de compte utilisateur	- Administrateur	- SCVRA
Msg20	Ajouter une couleur pour la voiture 3D	- Expert	- SCVRA
Msg21	Couleur ajoutée	- SCVRA	- Expert

Tableau C.1 : *Messages acteurs/ système.*

B. Diagramme du contexte

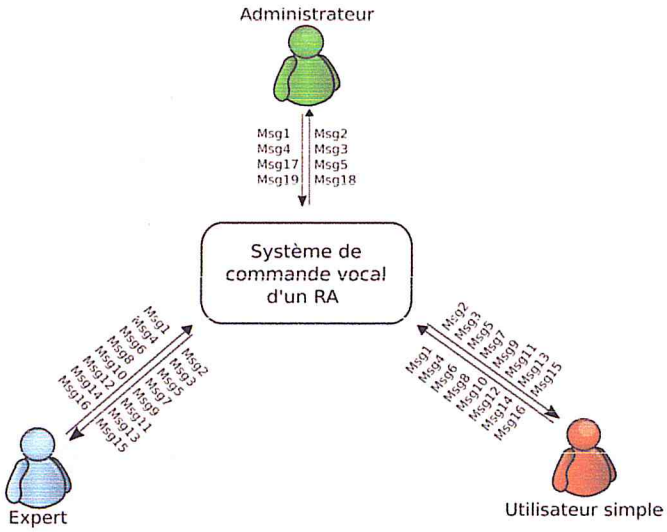


Figure C.4 : *Diagramme du contexte.*

C.4.2. Capture des besoins techniques

C.4.2.1 Identification des cas d'utilisation techniques

Le tableau ci-dessous, donne un aperçu des futures fonctionnalités techniques, que doit implémenter notre système :

Cas d'utilisation	Acteur	Messages émets /reçus
S'authentifier	- Administrateur - Utilisateur Simple - Expert	**Reçoit :** • Formule d'authentification **Emet :** • Remplir la formule **Reçoit :** • Autoriser l'accès à la session personnelle **Ou** • Refuser l'accès • Message d'erreur : login ou mot de passe incorrect

Etablir les préférences	- Administrateur - Utilisateur Simple - Expert	**Emet :** • Etablissement des préférences **Reçoit :** • Session modifiée selon les préférences
Gérer les sessions	Administrateur	**Emet :** • Demande de mise à jour d'un compte utilisateur **Reçoit :** • interface de mise à jour d'un compte utilisateur **Emet :** • Mise à jour d'un compte utilisateur **Reçoit :** • Message d'information : opération effectuée **Ou :** • Message d'erreur : opération refusée

Tableau C.2 : *Identification des cas d'utilisation techniques.*

C.4.2.2. Description détaillée des cas d'utilisations techniques :

A. S'authentifier :

Sommaire d'identification :

Titre : S'authentifier

But : Pour accéder à la session personnelle d'utilisateur.

Résumé : L'utilisateur doit saisir son login et mot de passe afin de s'authentifier.

Acteurs : Administrateur. Expert, Utilisateur Simple.

Description des enchainements :

Pré-conditions :

- L'utilisateur doit être inscrit.

Scénario nominal : ce cas d'utilisation commence lorsque l'utilisateur demande au système de s'authentifier :

- L'utilisateur saisit son login et mot de passe.
- Le système vérifie les informations saisies, si elles sont correctes, il ouvrir l'interface personnelle.

Enchainement alternatif:

- Le système va déclencher une erreur si :

- L'utilisateur saisit un login ou mot de passe faux.
- Le champ du login ou mot de passe est vide.

➢ L'utilisateur peut annuler à tout moment cette opération.

Exception :

➢ Erreur d'accès à la base de données.

Post-conditions :

➢ Accéder à l'interface personnelle.

B. Etablir les préférences :

Sommaire d'identification :

Titre : Etablir les préférences.
But : Pour rendre l'interface de l'utilisateur conviviale et ergonomique.
Résumé : L'utilisateur peut changer les paramètres de son interface et lancer le serveur.
Acteurs : Administrateur, Expert, utilisateur simple.

Description des enchainements :

Scénario nominal : ce cas d'utilisation commence lorsque l'utilisateur demande au système d'établir ses préférences :

➢ L'utilisateur peut sélectionner la résolution d'écran à utiliser.
➢ L'utilisateur active le serveur.

Enchainement alternatif:

➢ L'utilisateur peut annuler à tout moment ces opérations.

Post-conditions :

➢ Session modifiée selon les préférences de l'utilisateur.

C. Gérer les sessions

Sommaire d'identification :

Titre : Gérer les sessions.
But : Pour ajouter, modifier ou supprimer un compte utilisateur.
Résumé : La gestion des sessions, permet d'ajouter/modifier ou supprimer un compte utilisateur.
Acteurs : Administrateur.

Description des enchainements :

Pré-conditions :

- L'administrateur doit s'authentifier.

Scénario nominal : ce cas d'utilisation commence lorsque l'administrateur demande au système de :

a) Ajouter une nouveau compte d'utilisateur dans ce cas l'administrateur doit :

- Remplir la formule d'ajout.
- Valider l'ajout.

b) Supprimer un compte d'utilisateur, dans ce cas l'administrateur doit :

- Saisir le nom d'utilisateur.
- Valider la suppression.

c) Modifier un compte d'utilisateur, dans ce cas l'administrateur doit :

- Saisir le nom d'utilisateur.
- Modifier les informations.
- Valider la modification.

Enchainement alternatif:

- Le système déclenche une erreur, si l'administrateur ajoute un compte qui existe dans le système.
- Le système déclenche une erreur, si l'administrateur saisie des informations incorrectes.
- L'administrateur peut annuler à tout moment ces opérations.

Exception :

- Erreur d'accès à la base de données.

Post-conditions :

- Compte ajouté avec succès.
- Compte supprimé avec succès.
- Compte modifié avec succès.

C.4.3. Capture des besoins fonctionnels et des besoins d'interaction

C.4.3.1. Identification des cas d'utilisation fonctionnels

Nous allons donner dans ce qui suit la version finale des cas d'utilisation obtenus après plusieurs itérations :

Cas d'utilisation	Acteurs	Messages émets / reçus	Interaction
Afficher les composants 3D par famille	- Utilisateur Simple - Expert	**Emet :** • Demander les composants 3D **Reçoit :** • Des composants 3D **Ou** Message d'erreur : Affichage des familles par ordre	**Objets réel :** - Marqueur **Objets Virtuels :** - Composants 3D **Taches :** commande vocale pour afficher les composants 3D.
Mettre à jour d'un composant 3D	- Expert	**Emet :** • Demander la mise à jour d'un Composant 3D **Reçoit :** • Afficher les composants 3D **Emet :** • Sélectionner un composant 3D **Reçoit :** • Afficher les caractéristiques du composant 3D **Emet :** • Mise à jour du composant 3D **Reçoit :** • Message d'information : Opération effectuée **Ou** • Message d'erreur : Opération refusée	**Objets réel :** - Marqueur **Objets Virtuels :** - Composants 3D **Taches :** Click bouton pour cette tache
Modifier la couleur de voiture 3D	- Expert	**Reçoit :** • les composants 3D / voiture 3D **Emet :** • Modifier la couleur de voiture 3D **Reçoit :** • Message d'information : Opération effectuée **Ou** • Message d'erreur :	**Objets réel :** - Marqueur **Objets Virtuels :** - Composants 3D **Taches :** commande vocale pour modifier la couleur de la voiture 3D.

		Opération refusée	
Assembler un composant 3D	- Utilisateur Simple - Expert	**Emet :** • Demander l'assemblage d'un Composant 3D **Reçoit :** • Composant 3D assemblé **Ou** • Message d'erreur : Opération refusée	**Objets réel :** - Marqueur **Objets Virtuels :** - Composants 3D **Taches :** commande vocale pour assembler un composant 3D.
Sélectionner ou désélectionner un composant 3D	- Utilisateur Simple - Expert	**Reçoit :** • Afficher Les composants 3D **Emet :** • Sélectionner ou désélectionner un composant 3D **Reçoit :** • Composant 3D sélectionné ou désélectionné **Ou** Message d'information : Le composant n'existe pas.	**Objets réel :** - Marqueur **Objets Virtuels :** - Composants 3D **Taches :** commande vocale pour sélectionner ou désélectionner un composant 3D.
Manipuler un composant 3D/une voiture 3D (Rotation, translation, Zoom)	- Utilisateur Simple - Expert	**Emet :** • Demander une manipulation de composant 3D **Reçoit :** • Composant 3D/ voiture 3D manipulé(e).	**Composants réel :** - Marqueur **Composants Virtuels :** - Composants 3D - Voiture 3D **Taches :** commande vocale pour manipuler un composant 3D ou une voiture 3D. **OU** Click bouton : pour manipuler un composant 3D ou une voiture 3D.

Ajouter une couleur pour la voiture 3D.	-Expert	**Emet :** • Demander l'ajout d'une couleur de voiture 3D **Reçoit :** • Afficher l'interface d'ajout des couleurs **Emet :** • Ajout de la couleur. **Reçoit :** • Message d'information : Opération effectuée **Ou** • Message d'erreur : Opération refusée	**Taches :** Click bouton : pour ajouter une couleur pour la voiture 3D.

Tableau C.3 : *Identification des cas d'utilisation fonctionnels.*

C.4.3.2. Description détaillée des cas d'utilisation fonctionnels/interactionnels

A. Afficher les composants 3D par famille

Sommaire d'identification :

Titre : Afficher les composants 3D par famille.

But : Afficher les composants 3D par famille.

Résumé : L'utilisateur peut s'afficher une famille des composants 3D.

Acteurs : Expert, Utilisateur Simple.

Description des enchainements :

Pré-conditions :

- L'utilisateur doit s'authentifier.

Scénario nominal : ce cas d'utilisation commence lorsque l'utilisateur demande au système d'afficher les composants 3D par famille, par une commande vocale.

Enchainement alternatif:

- L'utilisateur peut annuler cette opération.

Exception :

- Des commandes vocales incorrectes.

Post-conditions :

Affichage des composants 3D par famille dans la scène.

B. Sélectionner ou désélectionner un composant 3D

Sommaire d'identification :

Titre : Sélectionner un ou désélectionner un composant 3D.

But : Pour sélectionner ou désélectionner un composant 3D.

Résumé : L'utilisateur peut sélectionner ou désélectionner un composant 3D.

Acteurs : Expert, Utilisateur Simple.

Description des enchainements :

Pré-conditions :

- L'utilisateur doit s'authentifier.
- Les composants 3D.

Scénario nominal : ce cas d'utilisation commence lorsque l'utilisateur demande au système de :

- Sélectionner un composant pour lui appliquer l'opération d'assemblage.

Ou

- Désélectionner un composant 3D si le composant est déjà sélectionné.

Enchainement alternatif:

- L'utilisateur peut annuler à tout moment cette opération.

Exception :

- Les commandes vocales incorrectes.
- Désélectionner un composant 3D qui n'est pas sélectionné.
- Le composant 3D n'existe pas.

Post-conditions :

Sélectionner ou désélectionner un composant 3D.

C. Modifier la couleur de la voiture 3D

Sommaire d'identification :

Titre : Modifier la couleur de la voiture 3D.

But : Pour modifier la couleur de la voiture 3D

Résumé : L'utilisateur peut modifier la couleur de la voiture 3D.

Acteurs : Expert, Utilisateur Simple.

Description des enchainements :

Pré-conditions :

- L'utilisateur doit s'authentifier.
- Les composants 3D existent dans la scène.

Scénario nominal : ce cas d'utilisation commence lorsque l'utilisateur demande au système de modifier la couleur d'une voiture 3D.

Enchainement alternatif:

- L'utilisateur peut annuler à tout moment cette opération.

Exception :

- Les commandes vocales sont incorrectes.
- La couleur n'existe pas.
- Le composant 3D n'existe pas.

Post-conditions :

La couleur de la voiture 3D est modifiée.

D. Manipuler un composant 3D/voiture 3D (Rotation/Zoom/Translation)

Sommaire d'identification :

Titre : Manipuler un composant 3D/voiture 3D (Rotation/Zoom).

But : Pour manipuler un composant 3D (soit une rotation, une translation ou bien un zoom).

Résumé : L'utilisateur peut manipuler un composant 3D.

Acteurs : Expert, Utilisateur Simple.

Description des enchainements :

Pré-conditions :

- L'utilisateur doit s'authentifier.
- Un composant 3D existe.

Scénario nominal : ce cas d'utilisation commence lorsque l'utilisateur :

- Prononce une commande vocale pour appliquer une rotation de composant 3D/ voiture 3D.

OU

- Utilise les touches clavier pour appliquer une rotation de composant 3D/voiture 3D.

- Utilise les touches clavier pour appliquer une «zoom» de composant 3D/voiture 3D.
- Utilise les touches clavier pour applique une «translation» de composant 3D/ voiture 3D.

Enchainement alternatif:

- L'utilisateur peut annuler à tout moment cette opération.

Exception :

- Des commandes vocales incorrectes.
- Le composant 3D n'existe pas.

Post-conditions :

- Le composant est tourné avec succès.
- Le composant est zoomé avec succès.
- le composant est déplacé avec succès.

E. Assembler des composants 3D

Sommaire d'identification :

Titre : Assembler des composants 3D.
But : Pour assembler des composants 3D.
Résumé : Un utilisateur assemble des composants 3D à la carcasse de voiture 3D.
Acteurs : Expert, Utilisateur Simple.

Description des enchainements :

Pré-conditions :

- L'utilisateur doit s'authentifier.
- Un composant 3D existe.
- Un composant initial existe (la carcasse de voiture 3D).
- Respecte l'ordre d'assemblage.

Scénario nominal : ce cas d'utilisation commence lorsque l'utilisateur demande au système d'assembler les composants 3D. L'utilisateur prononce une commande vocale pour assembler les composants 3D avec le composant initial (carcasse de voiture).

Enchainement alternatif:

- L'utilisateur peut annuler cette opération.

Exception :

- Des commandes vocales incorrectes.
- Le composant 3D n'existe pas.

Post-conditions :

- Les composants 3D sont assemblés avec succès.

F. Mettre à jour un composant 3D

Sommaire d'identification :

Titre : Mettre à jour un composant 3D.

But : Pour ajouter, modifier ou supprimer un composant 3D.

Résumé : La mise à jour permet d'ajouter/modifier ou supprimer un composant 3D.

Acteurs : Expert.

Description des enchainements :

Pré-conditions :

- L'expert doit s'authentifier.

Scénario nominal : ce cas d'utilisation commence lorsque l'expert demande au système de :

a) Ajouter un nouveau composant, dans ce cas l'expert doit :

- Remplir la formule d'ajout.
- Valider l'ajout.

b) Supprimer un composant 3D, dans ce cas l'expert doit :

- Saisir le nom de composant 3D.
- Valider la suppression.

c) Modifier un composant 3D, dans ce cas l'expert doit :

- Saisir le nom de composant 3D.
- Modifier les caractéristiques de composant 3D.
- Valider la modification.

Enchainement alternatif:

- Le système déclenche une erreur, si l'expert ajoute un composant qui existe déjà dans le système.

- Le système déclenche une erreur, si l'expert saisi les caractéristiques incorrectes pour le composant 3D.
- L'expert peut annuler à tout moment ces opérations.

Exception :

- Erreur d'accès à la base de données.

Post-conditions :

- Le composant est ajouté avec succès.
- Le composant est supprimé avec succès.
- Le composant est modifié avec succès.

G. Ajouter une couleur pour la voiture 3D

Sommaire d'identification :

Titre : Ajouter une couleur pour la voiture 3D.
But : Pour ajouter une couleur pour la voiture 3D.
Résumé : Cette opération, permet d'ajouter une nouvelle couleur pour la voiture 3D.
Acteur : Expert.

Description des enchainements :

Pré-conditions :

- L'expert doit s'authentifier.

Scénario nominal : ce cas d'utilisation commence lorsque l'expert demande au système, d'ajouter une nouvelle couleur pour la voiture 3D, dans ce cas l'expert doit :

- Remplir la formule d'ajout.
- Valider l'ajout.

Enchainement alternatif:

- le système déclenche une erreur, si l'expert ajoute une couleur qui existe déjà dans le système.

Exception :

- Erreur d'accès à la base de données.

Post-conditions :

- Le composant 3D est ajouté avec succès.